



DBKDA 2011

The Third International Conference on Advances in Databases, Knowledge, and
Data Applications

January 23-28, 2011 - St. Maarten,
The Netherlands Antilles

DBKDA 2011 Editors

Friedrich Laux, Reutlingen University, Germany

Lena Strömbäck, Linköpings Universitet, Sweden

DBKDA 2011

Foreword

The Third International Conference on Advances in Databases, Knowledge, and Data Applications (DBKDA 2011) held on January 23-27, 2011 in St. Maarten, The Netherlands Antilles, continued a series of international events covering a large spectrum of topics related to advances in fundamentals on databases, evolution of relation between databases and other domains, data base technologies and content processing, as well as specifics in applications domains databases.

Advances in different technologies and domains related to databases triggered substantial improvements for content processing, information indexing, and data, process and knowledge mining. The push came from Web services, artificial intelligence, and agent technologies, as well as from the generalization of the XML adoption.

High-speed communications and computations, large storage capacities, and load-balancing for distributed databases access allow new approaches for content processing with incomplete patterns, advanced ranking algorithms and advanced indexing methods.

Evolution on e-business, ehealth and telemedicine, bioinformatics, finance and marketing, geographical positioning systems put pressure on database communities to push the 'de facto' methods to support new requirements in terms of scalability, privacy, performance, indexing, and heterogeneity of both content and technology.

We take this opportunity to thank all the members of the DBKDA 2011 Technical Program Committee as well as the numerous reviewers. The creation of such a broad and high-quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and efforts to contribute to the DBKDA 2011. We truly believe that, thanks to all these efforts, the final conference program consists of top quality contributions.

This event could also not have been a reality without the support of many individuals, organizations, and sponsors. We are grateful to the members of the DBKDA 2011 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that DBKDA 2011 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in database research.

We are convinced that the participants found the event useful and communications very open. The beautiful places of St. Maarten surely provided a pleasant environment during the conference and we hope you had a chance to visit the surroundings.

DBKDA 2011 Chairs

Friedrich Laux, Reutlingen University, Germany
Aris M. Ouksel, The University of Illinois at Chicago, USA
Lena Strömbäck, Linköpings Universitet, Sweden

DBKDA 2011

Committee

DBKDA Advisory Chairs

Friedrich Laux, Reutlingen University, Germany
Aris M. Ouksel, The University of Illinois at Chicago, USA
Lena Strömbäck, Linköpings Universitet, Sweden

DBKDA 2011 Technical Program Committee

Abdel Alboody, Université Paul Sabatier (UPS) - Toulouse III , France
Annalisa Appice, Università degli Studi di Bari, Italy
Jean-François Baget, INRIA, France
Nadia Bennani, Université de Lyon, France
Patrick Bosc, Enssat - Lannion, France
Chin-Chen Chang, Feng Chia University Taiwan, Taiwan
Farid Bourennani, University of Ontario Institute of Technology (UOIT), Canada
Martine Cadot, LORIA-Nancy, France
Michelangelo Ceci, University of Bari, Italy
Qiming Chen, HP Labs - Palo Alto, USA
Alfredo Cuzzocrea, Italian National Research Council / University of Calabria, Italy
Maria Del Pilar Angeles, Universidad Nacional Autonoma de Mexico - Del Coyoacan, Mexico
Cédric du Mouza, CEDRIC-CNAM, France
Jana Dvoráková, Comenius University-Bratislava, Slovakia
Gledson Elias, Federal University of Paraíba, Brazil
Victor Felea, "A. I. Cuza" University of Iasi, Romania
Daniela Grigori, University of Versailles, France
Stephan Grimm, FZI Karlsruhe, Germany
Ismail Hababeh, United Arab Emirates University -Al-Ain, UAE
Takahiro Hara, Osaka University, Japan
Pascal Hitzler, Wright State University-Dayton, USA
Tobias Hoppe, Ruhr-University of Bochum, Germany
Edward Hung, The Hong Kong Polytechnic University - Hong Kong, PRC
Chris Ireland, Open University, UK
Najla Sassi Jaziri, ISSAT Mahdia, Tunisia
Wassim Jaziri, ISIM Sfax, Tunisia
Nhien An Le Khac, University College Dublin, Ireland
Sadegh Kharazmi, RMIT University, Australia
Kyoung-Sook Kim, National Institute of Information and Communications Technology, Japan
Christian Kop, University of Klagenfurt, Germany
Friedrich Laux, Reutlingen University, Germany
Alain Lelu, University of Franche-Comté, France
Chunmei Liu, Howard University - Washington DC, USA
Corrado Loglisci, University of Bari, Italy
Shuai Ma, University of Edinburgh, UK

Gerasimos D. Marketos, University of Piraeus, Greece
Florent Masségria, INRIA - Sophia Antipolis, France
Elisabeth Métais, CEDRIC / CNAM - Paris, France
Yasuhiko Morimoto, Hiroshima University, Japan
Aris M. Ouksel, The University of Illinois at Chicago, USA
Panagiotis Papapetrou, Aalto University, Finland
Alexander Pastwa, Ruhr-Universität Bochum, Germany
Dusan Petkovic, University of Applied Sciences - Rosenheim, Germany
Mathieu Roche, LIRMM, Université Montpellier 2, France
Satya Sahoo, Wright State University, USA
Abhishek Sanwaliya, Indian Institute of Technology - Kanpur, India
Ismael Sanz, Universitat Jaume I - Castelló, Spain
M. Saravanan, Ericsson India Pvt. Ltd -Tamil Nadu, India
Idrissa Sarr, University Pierre et Marie Curie - Paris, France
Umberto Straccia, ISTI - Italian National Research Council - Pisa, Italy
Lena Strömbäck, Linköpings Universitet, Sweden
Raj Sunderraman, Georgia State University, USA
Raphaël Thollot, SAP / Ecole Centrale Paris, France
Amel Touzi, Tunis El Manar University, Tunisia
Jianwu Wang, San Diego Supercomputer Center /University of California, USA
Maribel Yasmina Santos, University of Minho, Portugal
Jin Soung Yoo, Indiana University - Purdue University/Fort Wayne, USA
Zurinahni Zainol, University of Hull, UK
Wenjie Zhang, University of New South Wales, Australia

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

Trusted Data in IBM's Master Data Management <i>Przemyslaw Pawluk, Jarek Gryz, Stephanie Hazlewood, and Paul van Run</i>	1
A Field Analysis of Relational Database Schemas in Open-source Software <i>Fabien Coelho, Alexandre Aillot, Samuel Pilot, and Shamil Valeev</i>	9
The Use of Data Cleansing in Mobile Devices <i>Maria del Pilar Angeles, Francisco Garcia-Ugalde, and David Alcudia-Aguilera</i>	16
Remote Comparison of Database Tables <i>Fabien Coelho</i>	23
An Approach for Distributed Streams Mining Using Combination of Naive Bayes and Decision Trees <i>Meng Zhang and Guojun Mao</i>	29
Systems Biology Warehousing: Challenges and Strategies toward Effective Data Integration <i>Thomas Triplet and Gregory Butler</i>	34
Studying the Impact of Partition on Data Reduction for Very Large Spatio-temporal Datasets <i>Nhien An Le Khac, Martin Bue, and M-Tahar Kechadi</i>	41
Hybrid DCT-CT Digital Image Adaptive Watermarking <i>bijan fadaenia and nasim zarei</i>	47
A Secure Database System using Homomorphic Encryption Schemes <i>Youssef Gahi, Mouhcine Guennoun, and Khalil El-Khatib</i>	54
SQRM: An Effective Solution to Suspicious Users in Database <i>Hua Dai, Xiaolin Qin, Guineng Zheng, and Ziyue Li</i>	59
Exploring the Essence of an Object-Relational Impedance Mismatch - A novel technique based on Equivalence in the context of a Framework <i>Christopher Ireland</i>	65
Symbolic Representation and Reasoning for Rectangles with Superposition <i>Takako Konishi and Kazuko Takahashi</i>	71
New fuzzy multi-class methode to train SVM classifier <i>Taoufik Guernine and Kacem Zeroual</i>	77

MAXCLIQUE Problem Solved Using SQL <i>Jose Torres-Jimenez, Nelson Rangel-Valdez, Himer Avila-George, and Loreto Gonzalez-Hernandez</i>	83
Managing and Processing Office Documents in Oracle XML Database <i>Sabina Petride, Asha Tarachandani, Nipun Agarwal, and Sam Idicula</i>	89
An Algorithm for Clustering XML Data Stream Using Sliding Window <i>Guojun Mao, Mingxia Gao, and Wenji Yao</i>	96
Societal View on Knowledge Representation and Management: A Case Study of an ICT Consulting Company <i>Cheng-Chieh Huang and Ching-Cha Hsieh</i>	102
Exploring Statistical Information for Applications-Specific Design and Evaluation of Hybrid XML storage. <i>Lena Stromback, Valentina Ivanova, and David Hall</i>	108
Transforming XPath Expressions into Relational Algebra Expressions With Kleene Closure <i>Yangjun Chen</i>	114
Large Software Component Repositories into Small Index Files <i>Marcos P. Paixao, Leila Silva, Tales Brito, and Gledson Elias</i>	122
Intelligent Database Flexible Querying System by Approximate Query Processing <i>Oussama Tlili, Minyar Sassi, and Habib Ounelli</i>	128
IMA: Identification of Multi-author Student Assignment Submissions Using a Data <i>Kathryn Burn-Thornton and Tim Burman</i>	136
A Representation of Certain Answers for Views and Queries with Negation <i>Victor Felea</i>	142
Formation of Triads in Mobile Telecom Network <i>Saravanan Mohan and Naren Krishna</i>	148
Ambients of Persistent Concurrent Objects <i>Suad Alagic and Akinori Yonezawa</i>	155
Modeling Temporal Databases and Temporal Constraints <i>Mohamed Mkaouar, Mohamed Moalla, and Rafik Bouaziz</i>	162
Optimal Query Operator Materialization Strategy for Hybrid Databases <i>Martin Grund, Jens Krueger, Matthias Kleine, Alexander Zeier, and Hasso Plattner</i>	169
From Synchronous Corpus to Monitoring Corpus, LIVAC: The Chinese Case	175

Benjamin K. Tsou, Andy C. Chin, and Oi Yee Kwong

Efficient Access to Non-Sequential Elements of a Search Tree 181
Lubomir Stanchev

An Optimistic Transaction Model for a Disconnected Integration Architecture 186
Tim Lessner, Fritz Laux, Thomas Connolly, Cherif Branki, Malcolm Crowe, and Martti Laiho

A Concept for a Compression Scheme of Medium-Sparse Bitmaps 192
Andreas Schmidt and Mirko Beine

Merging Differential Updates in In-Memory Column Store 196
Jens Krueger, Martin Grund, Johannes Wust, Alexander Zeier, and Hasso Plattner

Trusted Data in IBM's Master Data Management

Przemyslaw Pawluk, Jarek Gryz
 York University
 Toronto ON, Canada
 Center for Advanced Studies
 IBM, Toronto, ON Canada
 Email: {pawluk, jarek}@cse.yorku.ca

Stephanie Hazlewood, Paul van Run
 IBM Laboratory
 Toronto ON, Canada
 Email: {stephanie, pvanrun}@ca.ibm.com

Abstract—A good business data model has little value if it lacks accurate, up-to-date customer data. This paper describes how data quality measures are processed and maintained in *IBM InfoSphere MDM Server* and *IBM InfoSphere Information Server*. It also introduces a notion of *trust*, which extends the concept of data quality and allows businesses to consider additional factors, that can influence the decision making process. The solutions presented here utilize existing tools provided by IBM in an innovative way and provide new data structures and algorithms for calculating scores for persistent and transient quality and trust factors.

Keywords-Master Data Management; Data Integration; Data Quality; Data Trust

I. INTRODUCTION

Many organizations have come to the realization that they do not have an accurate view of their business-critical information such as customers, vendors, accounts, or products. As new enterprise systems are added, silos are created resulting in overlap and inconsistency of information. This varied collection of systems can be the result of systems introduced through mergers and acquisitions (M&A), purchase of packaged applications for enterprise resource planning (ERP) or customer relationship management (CRM), different variant and versions of the same application used for different lines of business or home grown applications. Data in these systems typically differs both in structure and in content. Some data might be incorrect, some of it might just be old, and some other parts of it might show different aspects of the same entity (for example, a home vs. a work address for a customer).

Master Data Management (MDM) is an approach that decouples master information from the applications that created it and pulls it together to provide a single, unified view across business processes, transactional and analytical systems. Master data is not about all of the data of an organization. It is the data that deals with the core facts about the key entities of a business: customers, accounts, locations and products. Master data is high value data that is commonly shared across an enterprise – within or across the lines of business. MDM applications, such as IBM's InfoSphere Master Data Management Server, contain functionality to maintain master data by addressing key data issues such as governance, quality and consistency. They maintain and leverage relationships between master data entities and manage the complete lifecycle of the

data and support multiple implementation approaches.

The quality of master data requires special attention. Different aspects or dimensions of quality need to be considered and maintained in all processes of the enterprise. Trust scores, introduced in this paper, can provide important information to the decision makers. Our approach to the quality of data is slightly different than described so far in the literature [1]–[3]. Our goal is to provide the user with the information about data quality and trust. Trust in this case is the aggregated value of multiple factors, and is intended to cover quality and non-quality aspects of master data. We are not making the attempts to build fixes nor enforce any quality policy. The information provided by us is intended to identify weaknesses of data quality. The data quality enforcement should be then improved based on this information.

This paper focuses on the creation of measures, or trust factors, that serve to determine the trustworthiness of data being managed by MDM applications, specifically those being introduced in IBM's InfoSphere MDM Server. This new notion involves creating *trust scores* for these trust factors that enhance the notion of data quality and the more broad quality-unrelated features such as lineage, security, stewardship etc. All these have one goal – to support businesses in the decision making process, or data stewardship by providing information about different aspects of data.

This work is organized as follows. Section III presents the underpinning principles of Master Data Management (MDM), related concepts as well as the tools we used to prepare the trust scoring prototype. In Section IV we introduce a sample business scenario through which we explain the main ideas in the paper. Section V provides a short overview of data quality and introduces the notion of trust. Section VI presents structures and methods used to acquire, store and process trust factors.

II. BACKGROUND AND RELATED WORK

Data Quality has been explored by several researchers in recent years and its importance has been discussed many times in the literature [1]–[14] usually in context of the single data source. However, some researches have been also done in the context of integrated data [15]–[20]. Most of them present strictly theoretical approach to the topic and provides solutions that are hard to apply or expensive. Moreover, all of them

includes only quality factors into the considerations excluding several extremely important non-quality factors such as data lineage or security. Different approaches to data quality and chosen definitions are presented more detailed in Section V.

In this work we would like to extend the notion of data quality and introduce new notion called *data trust* which covers both quality and non-quality factors. We propose a set of tools that can be used to process this information.

III. MDM AND INFORMATION SERVER

Master data management is a relatively fast growing software market. Many customers acknowledge they have data quality and data governance problems and look to large software vendors like IBM for solutions to these problems. Crucial parts of these MDM solutions are data quality and data trust mechanisms [21]–[23]. In this section, we are presenting the MDM environment and the comprehensive approach to the trust and quality that utilizes tools provided by IBM.

A. Definitions

Master Data Management (MDM) provides the technology and processes to manage Master Data in an organization. Master Data is the data an organization stores about key elements or business entities that define its operation. "An MDM solution enables an enterprise to govern, create, maintain, use, and analyze consistent, complete, contextual, and accurate master data information for all stakeholders, such as line of business systems, data warehouses, and trading partners." [21] Master data is high value information that an organization uses repeatedly across many business processes and lines of businesses. For these to operate efficiently, this master data must be accurate and consistent to ensure good decisions. Unfortunately in many organizations, master data is fragmented across many applications, with many inconsistent copies and no plan to improve the situation.

Traditional approaches to master data include the use of existing enterprise applications, data warehouses and even middleware. Some organizations approach the master data issue by leveraging dominant and seemingly domain-centric applications, such as a customer relationship management (CRM) application for the customer domain or an enterprise resource planning (ERP) application for the product domain. However, CRM and ERP, among other enterprise applications, have been designed and implemented to automate specific business processes such as customer on-boarding, procure-to-pay and order-to-cash, not to manage data across these processes. The result is that a specific data domain, such as customer or product, may actually reside within multiple processes, and therefore multiple applications.

In general, master data management (MDM) solutions should offer the following:

- Consolidate data locked within the native systems and applications
- Manage common data and common data processes independently with functionality for use in business processes

- Trigger business processes that originate from data change
- Provide a single understanding of the domain-customer, product, account, location — for the enterprise

Depending on MDM tool those requirements are realized in a different way. Some products decouple data linked to source systems so they can dynamically create a virtual view of the domain, while others include the additional ability to physically store master data and persist and propagate this information. Some products are not designed for a specific usage style, while others provide a single usage of this master data. Even more mature products provide all of the usage types required in today's complex business-collaborative, operational and analytic-as out-of-the-box functionality. These mature products also provide intelligent data management by recognizing changes in the information and triggering additional processes as necessary. Finally, MDM products vary in their domain coverage, ranging from specializing in a single domain such as customer or product to spanning multiple and integrated domains. Those that span multiple domains help to harness not only the value of the domain, but also the value between domains, also known as relationships. Relationships may include customers to their locations, to their accounts or to products they have purchased. This combination of multiple domains, multiple usage styles and the full set of capabilities between creating a virtual view and performance in a transactional environment is known as multiform master data management.

Some of the most common key business drivers for MDM are:

- Revenue Enhancement - More intelligent cross-sell and up-sell via complete understanding of customer (profile, accounts and interactions) to leverage bundling opportunities;
- Consistent Customer Treatment - Blending channels to deliver common customer interactions/experiences across all touch points;
- Operational Savings and Efficiencies - "Once & done" enterprise-wide services for key customer processes such as account changes (name, address);
- Privacy & Regulatory Compliance - Central location for consistent rules of visibility & entitlements;
- M&A Infrastructure - Shortening M&A customer, desktop, and billing integration time frames;

Achieving a high level of data quality is key prerequisite for many of the MDM objectives. Without high quality data the best analytics and business intelligence applications are still going to deliver unreliable input to important business decisions. Another key aspect of the management of the master data is achieving a high level of trustworthiness in the data. It is a key factor for customers to have reliable information about the data. Information about the quality, the origin, the timeliness and many other factors influencing the business decisions based on the provided data.

The introduction of *data governance* in the organization is a

vital prerequisite to come to more trusted information. Moving to master data management can be the cornerstone of a data governance program. It is important however, to note that at the same time, moving to MDM cannot be successful without data governance.

Data governance is defined as "the orchestration of people, process and technology to enable an organization to leverage information as an enterprise asset" [24]. It manages, safeguards, improves and protects organizational information. The effectiveness of data governance can influence the quality, availability and integrity of data by enabling cross-organizational collaboration and structured policy-making.

B. MDM Tools

IBM InfoSphere MDM Server is an application that was built on open standards and the Java Enterprise Edition (JEE) platform. It is a real-time transactional application with a service-oriented architecture that has been built to be scalable from both volume and performance perspectives. Shipping with a persistent relational store, it provides a set of predefined entities supporting the storage of master data applicable to each of the product's predefined domains.

This product also includes the MDM Workbench – an integrated set of Eclipse plug-ins to IBM Rational Software Architect/Developer that support the creation of new MDM entities and accompanying services, and a variety of extensions to MDM entities. This tooling reduces the time and breadth of skills required for solution development tailored to the business and allows for flexibility to changing business requirements with its model-driven approach to solution development. We also use the new module of the IBM Information Server Suite, *IBM InfoSphere Information Analyzer* that profiles and analyzes data so that the system can deliver trusted information to users. The Information Analyzer (IA) will be used to scan or sample data in data sources to assess the quality. It enables us to discover the structure of the quality and to give some guidelines how it can be improved. We are also using the complementary tools which are: *IBM InfoSphere QualityStage*, which allows us to define rules to standardize and match free-form data elements which is essential for effective probabilistic matching of potentially duplicate records, and *IBM WebSphere AuditStage*, which enables us to apply professional quality control methods to manage the accuracy, consistency, completeness, and integrity of information stored in databases. We also use statistics provided by *IBM InfoSphere DataStage* to compute chosen quality and trust factors.

This set of tools enables us to create the comprehensive approach to the data quality and data trust management. This approach not only resolves some problems during the data acquisition but also allows us to control the level of data trust and to give up-to-date information about the trustworthiness to the user. This comprehensive approach is novel. Moreover our solution does not require any specialized hardware or operating system and is able to cooperate with many commercial data base solution like DB2, Oracle and others.

1) *IBM InfoSphere MDM Server*: The InfoSphere Master Data Management Server has a new feature allowing users to define and add quality and trust factors to the data of their enterprise. This new data structure enables the user to store data required to compute scorings for trust and quality of data. Provided wizards allow the user to modify the data model in a simple way.

2) *IBM Information Server*: IBM Information Server addresses the requirements of cooperative effort of experts and data analysts with an integrated software platform that provides the full spectrum of tools and technologies required to address data quality issues [25]. It supplies users and experts with the tooling that allows the detailed analysis of data through profiling (IBM InfoSphere Information Analyzer and IBM InfoSphere AuditStage), cleansing (QualityStage) and data movement and transformation (DataStage). In this paper we concentrate on data profiling and analysis. Our work is focused mostly on IBM InfoSphere Information Analyzer (IA), IBM InfoSphere AuditStage (AS), and partially on QualityStage (QS).

IA, as an important tool of *data quality assessment* (DQA) process, aids the exposing technical and business issues. The technical issues detection is a simpler part of the process based on technical standards and covers following problems:

- Different or inconsistent standards in structure, format, or values
- Missing data, default values
- Spelling errors
- Data in wrong fields
- Buried information in free-form fields

Business quality issues are more subjective and are associated with business processes such as generating accurate reports. They require the involvement of the experts. IA helps the expert in systematic analysis and reporting of results, thereby allowing him to focus on the real problem of data quality issues. This is done through the following tasks:

Column Analysis – can be performed on all the columns of one or more tables, or selectively on certain columns and allows to run an analysis on the full volume of data, or on a subset using a sampling technique. As a result of this process reference tables can be generated. It enables later use of this information to determine the trustworthiness of data and as an input for data quality improvement process.

Key Analysis – IA offers two type of analysis *Primary Key Analysis* (PKA) and *Foreign Key Analysis* (FKA). PKA identifies primary keys, if not defined, and validate already defined keys. It is an important analysis in terms of duplicates detection and uniqueness verification. The second task (FKA) is defined to determine undefined, and validate defined, relationships between tables. The foreign key analysis job builds a complete set of all the column pairs between the primary key columns and the remaining selected columns in the selected tables. The primary key column of one table is paired with all of the columns of the other tables. Next, the system performs a compatibility test on each column pair to determine whether those columns are compatible with each other. If the

column pair is compatible, the columns are flagged and then evaluated further. Columns are considered compatible when format, length, scale, and precision matches. After reviewing the results of the job, user can test for referential integrity and determine if a foreign key candidate should be selected as a foreign key.

Cross-Table Analysis – called also cross-domain analysis, is used to determine whether columns contain overlapping or redundant data. It compares the data values between two columns to locate overlapping data. This type of analysis is a multiple step process which contains following steps:

- Selection of two or more columns
- Run a cross-domain analysis job – a list of all of the possible column pairs in data is generated.
- Compatibility test on each column pair to determine whether those columns are compatible (the same test is performed in FKA).

After the compatibility test, cross-domain analysis displays the results from the compatibility test for the user to review and optionally mark a column redundant.

a) *QualityStage*: IBM InfoSphere QualityStage (QS) complements IA by investigating free-form text fields such as names, addresses, and descriptions. QS allows user to define rules for standardizing free-form text domains which is essential for effective probabilistic matching of potentially duplicate master data records. QS provides user with a set of functionalities containing functions such as free-form text investigation, standardization, address verification and record linkage and matching as well as survivorship that allows best data across different sources to be merged.

b) *AuditStage*: IBM WebSphere AuditSatge (AS) enables user to apply professional quality control methods to manage different subjective quality factors of information stored in databases such as accuracy, consistency or completeness. By employing technology that integrates Total Quality Management (TQM) principles with data modeling and relational database concepts, AS diagnoses data quality problems and facilitates data quality improvement effort. It allows performing assessment of the completeness, validity of critical data elements and business rule compliance. User can evaluate the quality of data in terms of specific business rules involving multiple data fields within or across records (or rows) that are logically related. In most cases, the type of business rules needed for business rule analysis will not be documented or even explicitly known before the evaluation begins. Therefore, business rules applicable to data will need to be developed, or at least refined, for this analysis [25]. Sources of that knowledge are:

- knowledgeable people (subject matter experts),
- system documentation, and
- occasionally metadata repositories.

AuditStage is very useful tool for assessment of the factor called consistency allowing cross-table rules validation.

IV. WORKING EXAMPLE

Consider a typical scenario in an insurance industry. Insurance companies store information about entities including Customer, which can be a person or an organization and Contracts (variety of insurance policies i.e. home, life or car insurance). The company has to keep some information about employees.

MDM Server supports businesses providing predefined data models, containing many of the essential entities for storing this information. One can add additional attributes to entities in this predefined model using the MDM Workbench to generate so called data extensions.

Once the data domain has been defined, the next step is to impose constraints through rule generation. Those rules may belong to one from the following groups:

- Formatting rules – describing different formatting issues like length, allowed characters etc.
- Integrity constraints – rules describing i.e multiplicity of relations
- Business rules – any other rule i.e. dependencies among different fields and values

Table I shows a few possible rules identified for our example. In practice the number of rules generated and stored can be enormous, from 800 rules when assigning a claim, up to 1800 rules applied when underwriting the insurance policy¹.

V. TRUST NOTION

Trust is an extension of data quality. Data quality is not the only factor influencing the trustworthiness of data and these two concepts are not necessarily correlated. Low-quality data may be considered to have high trust and vice versa. The value of trust strongly depends on the user requirements and usage context. In this section, we discuss data quality and introduce the notion of trust.

A. Data Quality

The concept and importance of DQ has been discussed many times in the literature [1]–[14] usually in context of the single data source. However, some research has been also done in the context of integrated data [15]–[20] emphasizing the importance of data quality assurance in this context. In [1] there are three examples of organizational processes where DQ aspects are extremely important.

- Customer matching – it is a common issue in organizations where more than one system with overlapping databases exists. In such case issues with synchronization appear resulting in inconsistent and duplicate information.
- Corporate house-holding – is a problem of identifying members of household (or related group). This context-dependent issue is widely described in [26].
- Organization fusion – is the issue of integration legacy software in case of organizations or units merge.

Many different definitions of DQ can be found in literature. Some of them concentrate on intrinsic values such as accuracy

¹Based in internal IBM materials provided by ILOG team

TABLE I
RULES IDENTIFIED IN THE SAMPLE DOMAIN

No	Name	Description
Format		
F.1	Surname length	The length of surname should be at least 2 signs and at most 30 signs
F.2	Name length	The length of name should be at least 2 signs and at most 30 signs
Integrity Constraints		
I.1	Policy date and birth date of policy holder	The birth date of the policy holder must be earlier than policy start date
I.2	Claim date	Claim may be done only during the coverage period. Claim date must be later than policy start date and earlier than policy end date
Business Rules		
B.1	Currency	Data should be verified at least once in five years (60 months). The value of currency factor is equal to $1 - \frac{60}{months(current_date - last_verified_dt)}$
B.2	Policy Holder min age	The minimal age of policy holder is 18
B.3	Replaced contract id	Replaces_contract contains NULL or id of other contract which has been replaced

[8] and completeness [27] and not consider data in a context. Others try to compute values based on some usage context [11]. Instead of single definition, DQ is often split into dimensions or factors – metrics which are more formally defined and can be used measure and compare quality of data sets. But even then the same feature may be called differently by two researchers. This problem has been noticed by Wang and Strong [13] and Foley and Helfert [7].

Naumann [19] attempts to provide operational definition of DQ as "an aggregated value of multiple IQ-criteria" (Information Quality Criteria). IQ-criteria are there classified into four sets:

- Content-related – intrinsic criteria, concerning the retrieved data,
- Technical – criteria measuring aspects determined by software and hardware of the source, the network and the user,
- Intellectual – subjective aspects of data that shall be projected to the data in the source,
- Instantiation-related – criteria concerned on the presentation of the data.

We will follow the Naumann's approach by defining data quality as a aggregated value of multiple DQ-factors. Later we will extend this definition introducing the notion of trust.

B. Trust Definition

Following Naumann's definition of data quality, we define trust (data trust, DT) as *the aggregated value of multiple DT-factors*. This definition provides flexibility when defining trust for a specific industry and user requirements. The trust factor (DT-factor) may be a DQ-factor, as defined earlier in this section, or non-quality (NQ) factor. Here we will concentrate on NQ factors.

1) *Data Lineage*: Data lineage captures the ratings of data or data sources based on the origin and/or history of processing the data has been through. For example, some sources may be considered as more accurate than others. Information on how much data has been exposed to factors that may have caused

errors or inconsistencies (poorly secured systems, systems with poor error handling and checking) is important when considering how to calculate scores giving a measure of trust in the data.

a) *Origination*: is a factor that captures the scoring of the source of the data. Setting such rating requires the expertise and is strongly context/usage dependent. It may be used in situations where the information about origin is one of the key factors in decision making process.

b) *Traceability*: is an extension to the origination factor. It assesses the ability to trace the history of data. It gives us the information how much we know (or may know) about the previous places of storage and transformation done over the data element.

c) *Stewardship Status*: is the scoring capturing the stewardship assigned to the data. It assess if the data is managed manually or in some automatic, more or less limited, way by the system.

2) *Data Security*: This group of factors covers the security aspects of the systems storing data elements (now and in the past). The values of those factors can be assigned by the expert as well as using some tool that is able to run a security audit task over the system.

a) *Authentication*: is a scoring telling us how strong authentication mechanisms of the system are. It encloses, but is not necessarily limited to, permissions, password strength, password update policy etc.

b) *Authorization*: captures the strength of policies regulating the granting of access to the data and tasks in the system.

c) *Roles Policy*: concerns the aspects of roles management in the system i.e. using primary (root) roles and secondary roles that are limited.

d) *Auditing Policy*: captures the scores assessing the strength of auditing policies i.e. tracking dates and users initiating tasks. This kind of information may be crucial in the organizations operating on sensitive or confidential data.

3) *Trust of Data Sources*: The following factors capturing different aspects of data source trust [1]:

a) *Believability*: describes how true, real and credible a data source is.

b) *Reputation*: describes how trustable is the source. It is based on the experts' knowledge and is subjective.

c) *Objectivity*: defines the impartiality of source in data provisioning.

d) *Reliability*: is a factor describing whether a source provides data conveying the right information.

These definitions are not operational. Moreover they are qualitative and require transformation into quantitative measures to be applicable in our framework.

DT-factors on the row (entity) level are boolean values capturing the rules' satisfaction. On higher levels (table or query) they are expressed as a percentage of tuples satisfying the rule.

VI. TRUST PROCESSING

Trust and quality processing described below is one of the most novel aspects of our work. An important advantage of our approach is the use of existing set of tools, slightly modified or extended to serve in new context. We extend those tools by creating data structures to store and process metadata describing data quality and trust. We have implemented mechanisms for assessing some of the quality and trust factors.

A. Trust Data Structures

MDM provides a mechanism that enables an extension of the existing data with trust/quality factors. These extensions may be defined as *persistent object* and stored in the database or be *transient objects* calculated at run time.

MDM allows adding necessary classes and fields to the existing data model. We have used persistent fields as well as transient fields. Defined objects contain fields representing trust factors like age and volatility. Values of those fields are taken from database or calculated during the transaction's execution for the persistent and transient objects respectively. The acquisition process and the calculation methods are described in the following subsections.

Persistent objects are stored in the database. There are two possible solutions:

- Extension and extended object stored in the same table – the table is then extended by addition of new attributes.
- Extension is stored in a separate table – there is foreign key relation defined between the table storing extended data and table storing the extension.

In both cases while requesting the entity, there will be added information about the extension to the response.

B. Acquisition and Processing of Trust

In section IV, we have identified a set of rules that define quality requirements. Now, we will explain how these rules may be used to provide the information about quality of data.

1) *Transient factors*: MDM Server provide user with the ability of creating *behavioral extensions*. A behavior extension allows a client to plug in new business rules or functionality to work in conjunction with existing services or functionality within MDM Server. The following rule implements the rule B.1 from the Table I. It assigns the value of attribute *acrccy* according to this the rule B.1.

```
if years(current_date -
(last_verified_dt of the Person))
is more than 5
then
    set crncy to 0
else
    set crncy to
1-(months(current_date -
(last_verified_dt of the Person)) / 60)
```

The extension is executed when triggering event occurs, i.e. we can define the extension triggered by a select event done over Person. Before user receives Person, the system calls our extension and calculates transient trust factors accordingly, based on values stored in database. Priority enable us to define the order of calls.

MDM allows us to define different triggers for behavioral extensions:

- Action – Specifies component level transaction name. e.g. 'getPerson' or 'updateStudent'; each transaction is associated with a particular Module within MDM Server
- Transaction – Applies extension to a specific transaction at the controller level
- Action category – Specifies at the component level what category of transactions are to be impacted by the extension e.g. Add, Update, View, All
- Transaction category – Determines whether the extension will apply to a category of transactions e.g. inquiry, persistence or all at the controller level

The first two call an extension triggered by a specific action on chosen entity (i.e. updatePerson) on component or controller level respectively. Action and transaction category, on other hand allows the user to define the extension triggered by specific class of action or transaction, that is, to add or update done over any entity. In addition, extensions may be called before or after the action or transaction initiated by user.

2) *Persistent factors*: Values of scores for persistent factors are stored in the database. Data structures required to store this information has been defined as a *data extension* in the MDM Server. The acquisition of persistent factors can be off-line or on-line process. We use IA and AS to acquire trust scores. Those tools are used to acquire scorings in off-line mode. MDM Server allows us to modify persistent factors in the on-line mode. In such case we have to define the behavioral extension that is triggered by update or insert event. Figure 1 depicts dependencies among functions in Information Analyzer. Basically it defines order in which chosen functions may be called.

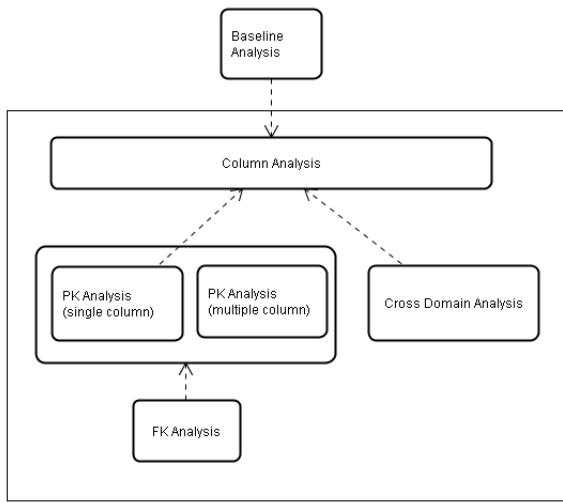


Fig. 1. IBM WebSphere Information Analyzer function dependencies

- Baseline Analysis requires at least column analysis to be performed before it can be run.
- Column Analysis must be run before a single column Primary Key Analysis can be performed.
- A multi-column Primary Key Analysis can be performed independently of any of the other analyses. It invokes Column Analysis automatically under the covers if a Column Analysis has not been performed on the selected columns.
- A Foreign Key Analysis (single or multi-column) can only be performed after a Primary Key Analysis (single or multi-column) is performed.
- Cross Domain Analysis requires Column Analysis to have been run.

Column analysis job evaluates data characteristics by analyzing a frequency distribution of the data values in each column. It is performed to evaluate the characteristics such as: minimum, maximum and average field length, precision, data types, cardinality, nullability and empty values. This task gives the structural view of data and returns inferences (best choices that system could make) in terms of field's length, data class, and uniqueness of values or constants that may indicate unused columns.

While IA is a tool to measure basic aspects of data quality (formatting, data types, precision, etc.), AuditStage (AS) can be used to implement more complex rules i.e. business rules. Results of the execution of such predefined rules are stored in the database and used to determine the overall quality of data.

IA as well as AS, are both run as a scheduled batch jobs. The frequency of such operation depends on user requirements and domain. It is obvious that long time interval between two executions can jeopardize the reliability of results, however, it is expensive operation. One needs to make a tradeoff to minimize costs and maximize the reliability of the results.

Another rich source of information about DQ are results stored by DataStage. This tool, used in general to perform

transformations of data and transitions from source to target data source, produces statistics, as a side effect. This information about merged records or unmatched records, gives us very important input for trust computation. This information, processed by MDM's behavioral extensions provides user with the information about consistency among data sources and can also point to underlying problems with data, such as inconsistent data coming from two systems caused by i.e. incorrect matching.

C. Trust processing

The trust alone is just yet another piece of data given to the user. The really important question is *What can be done with this information?* Let's consider now some cases showing usage of the trust in the system.

We have shown that the trust score can be incorporated in our meta-data and linked with each field in the database if desired. This information can be then returned to the user. Even though this information is very detailed, it is not practically useful in all cases. Without algorithms to propagate trust in the query processing, we can only annotate a tuple and return it to the user. However, we can build some statistics over this information that can be used later.

One of the problems that are currently unsolved is propagation of trust scores in the query processing. We are currently working on methods allowing us to estimate the trust of the result of the SQL operator based on the estimated trust of entry set. We are using estimates in this context because it is significantly less expensive than reaching out each time for the data.

There are many interesting problems in this domain. One of them is the impact of the trust of the key attributes on the trust of the result. This issue originates from the observation made by Motro and Rakov [28] that the measure can be considered accurate only if the key of the tuple is accurate. For example, when we consider the *group by* operation, there is significant influence of the group by keys on the trust to the aggregation result. It is intuitive especially in the context of accuracy dimension: simply if the group by key is highly inaccurate, then division into groups cannot be trusted. That leads to low level of trustworthiness of the aggregation result, even if the accuracy of the measure itself is high. Similar problems arise for join operation. However, in this case the accuracy of the keys of the join has to be propagated through the whole tuple, because inaccurate value of one of join components implies that the derived tuple should not be in the result set.

VII. CONCLUSION AND FUTURE WORK

Measuring data quality and data trust is one of the key aspects of supporting businesses in decision making process or data stewardship. Master Data Management in other hand supports sharing data within and across lines of business. In such case trustworthiness of the shared data is extremely important. Our investigation has resulted in consistent method of gathering and processing quality and trust factors.

In this work we have presented the *IBM InfoSphere MDM Server* and elements of *IBM Information Server* such as DataStage, QualityStage, AuditStage and Information Analyzer, and their ability to handle data quality and data trust. We have also presented the new notion of data trust. The process of gathering and computing data quality and trust factors has been described and explained using example.

At this point we would like to point to some aspects of MDM and DQ that have been mentioned in this work, but have not been covered in detailed. These aspects play important role in quality and trust computation. An extremely important feature in terms of defining business rules or any other rule and reusing them across cooperating systems is common rule repository and common rule engine. Those two elements can allow users to reuse defined rules and minimize probability of inconsistencies across systems. Such approach will also minimize costs because it eliminates a need to redefine rules in each system. Another aspect of trust and data governance not covered by this paper is temporal aspect of trust. In many cases trust strictly depends on time and a value i.e. address can be considered trustworthy only within a given time interval. This paper does not cover those aspects of quality and trust computation but we would like to point that there is ongoing work in IBM to solve this problem.

VIII. ACKNOWLEDGMENTS

We would like to thank Guenter Sauter for valuable discussions and uncovering new aspects of trust.

REFERENCES

- [1] C. Batini and M. Scannapieco, *Data Quality: Concepts, Methodologies and Techniques*, ser. Data-Centric Systems and Applications. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.
- [2] P. B. Crosby, *Quality is free : the art of making quality certain / Philip B. Crosby*. McGraw-Hill, New York :, 1979.
- [3] L. English, "Information quality improvement: Principles, methods, and management," Seminar, INFORMATION IMPACT International, Inc., 1996, 5th Ed., Brentwood, TN: INFORMATION IMPACT International, Inc.
- [4] D. Ballou and H. Pazer, "Modeling data and process quality in multi-input, multi-output information systems," *Management Science*, vol. 31, no. 2, pp. 150–162, 1985.
- [5] D. Ballou, R. Wang, H. Pazer, and G. K. Tayi, "Modeling information manufacturing systems to determine information product quality," *Manage. Sci.*, vol. 44, no. 4, pp. 462–484, 1998.
- [6] A. Parssian, S. Sarkar, and V. S. Jacob, "Assessing information quality for the composite relational operation join," in *IQ*, 2002, pp. 225–237.
- [7] O. Foley and M. Helfert, "The development of an objective metric for the accessibility dimension of data quality," in *Proceedings of International Conference on Innovations in Information Technology*. Dublin: IEEE, 2007, pp. 11–15.
- [8] J. Olson, *Data Quality: The Accuracy Dimension*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2002.
- [9] A. Parssian, S. Sarkar, and V. S. Jacob, "Assessing data quality for information products: Impact of selection, projection, and cartesian product," *Manage. Sci.*, vol. 50, no. 7, pp. 967–982, 2004.
- [10] T. C. Redman, *Data quality : the field guide*. Boston: Digital Pr. [u.a.], 2001.
- [11] G. K. Tayi and D. P. Ballou, "Examining data quality," *Commun. ACM*, vol. 41, no. 2, pp. 54–57, 1998.
- [12] A. R. Tupek, "Definition of data quality," U.S Department of Commerce, Census Bureau Methodology & Standards Council, 2006.
- [13] R. Y. Wang and D. M. Strong, "Beyond accuracy: what data quality means to data consumers," *J. Manage. Inf. Syst.*, vol. 12, no. 4, pp. 5–33, 1996.
- [14] L. P. Yang, Y. W. Lee, and R. Y. Wang, "Data quality assessment," *Communications of the ACM*, vol. 45, pp. 211–218, 2002.
- [15] Y. Cui, J. Widom, and J. L. Wiener, "Tracing the lineage of view data in a warehousing environment," *ACM Trans. Database Syst.*, vol. 25, no. 2, pp. 179–227, 2000.
- [16] M. Bouzeghoub and Z. Kedad, *Quality in Data Warehousing*. Kluwer Academic Publisher, 2002.
- [17] A. Gupta and J. Widom, "Local verification of global integrity constraints in distributed databases," in *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26-28, 1993*, P. Buneman and S. Jajodia, Eds. ACM Press, 1993, pp. 49–58.
- [18] M. Gertz and I. Schmitt, "Data Integration Techniques based on Data Quality Aspects," in *Proceedings 3. Workshop "Föderierte Datenbanken", Magdeburg, 10/11. Dezember 1998*, I. Schmitt, C. Türker, E. Hildebrandt, and M. Höding, Eds. Aachen: Shaker Verlag, 1998, pp. 1–19. [Online]. Available: citeseer.ist.psu.edu/19916.html
- [19] F. Naumann, *Quality-driven query answering for integrated information systems*. New York, NY, USA: Springer-Verlag New York, Inc., 2002.
- [20] M. P. Reddy and R. Y. Wang, "Estimating data accuracy in a federated database environment," in *CISMOD*, 1995, pp. 115–134.
- [21] A. Dreibelbis, E. Hechler, B. Mathews, M. Oberhofer, and G. Sauter, "Master data management architecture patterns," <http://www.ibm.com/developerworks/data/library/techarticle/dm-0703sauter/index.html>, 2007.
- [22] W. Fan, "Dependencies revisited for improving data quality," in *PODS '08: Proceedings of the twenty-seventh ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. New York, NY, USA: ACM, 2008, pp. 159–170.
- [23] J. Radcliffe and A. White, "Key issues for master data management," Gartner Master Data Management Summit, Chicago, IL, 2008.
- [24] IBM, "Ibm master data management: Effective data governance," <ftp://ftp.software.ibm.com/software/uk/itsolutions/information-management/information-transformation/master-data-management/master-data-management-governance.pdf>, 2007.
- [25] N. Alur, R. Joseph, H. Mehta, J. T. Nielsen, and D. Vasconcelos, *IBM WebSphere Information Analyzer and Data Quality Assessment*, ser. Redbooks. International Business Machines Corporation, 2007.
- [26] R. Y. Wang, K. Chettayar, F. Dravis, J. Funk, R. Katz-Haas, C. Lee, Y. Lee, X. Xian, and S. Bhansali, "Exemplifying business opportunities for improving data quality from corporate household research," in *Advances in Management Information Systems - Information Quality (AMIS-IQ) Monograph*, April 2005.
- [27] Y. W. Lee, L. Pipino, D. M. Strong, and R. Y. Wang, "Process-embedded data integrity," *J. Database Manag.*, vol. 15, no. 1, pp. 87–103, 2004.
- [28] A. Motro and I. Rakov, "Not all answers are equally good: estimating the quality of database answers," pp. 1–21, 1997.

A Field Analysis of Relational Database Schemas in Open-source Software

Fabien Coelho, Alexandre Ailios, Samuel Pilot, and Shamil Valeev

CRI, Maths & Systems, MINES ParisTech

35, rue Saint Honoré, 77305 Fontainebleau cedex, France.

fabien.coelho@mines-paristech.fr, firstname.lastname@mines-paris.org

Abstract—The relational schemas of 407 open-source projects storing their data in MySQL or PostgreSQL databases are investigated by querying the standard *information schema*, looking for various issues. These SQL queries are released as the *Salix* free software. As it is fully relational and relies on standards, it may be installed in any compliant database to help improve schemas. The overall quality of the surveyed schemas is poor: a majority of projects have at least one table without any primary key or unique constraint to identify a tuple; data security features such as referential integrity or transactional back-ends are hardly used; projects that advertise supporting both databases often have missing tables or attributes. PostgreSQL projects have a better quality compared to MySQL projects, and it is even better for projects with PostgreSQL-only support. However, the difference between both databases is mostly due to MySQL-specific issues. An overall predictor of bad database quality is that a project chooses MySQL or PHP, while good design is found with PostgreSQL and Java. The few declared constraints allow to detect latent bugs, that are worth fixing: more declarations would certainly help unveil more bugs. Our survey also suggests some features of MySQL and PostgreSQL as particularly error-prone. This first survey on the quality of relational schemas in open-source software provides a unique insight in the data engineering practice of these projects.

Keywords—open-source software; database quality survey; automatic schema analysis; relational model; SQL.

I. INTRODUCTION

In the beginning of the computer age, software was freely available, and money was derived from hardware only. Then in the 70s it was *unbundled* and sold separately. Stallman initiated the free software movement, which is now quite large [1], to implement his principle of sharing software. Such free software is distributed under a variety of licenses. The common ground is that it must be available as source code to allow its study, change and improvement, as opposed to compiled or obfuscated, hence the expression *open source*. This induces many technical, economical, legal, and philosophical issues. Open-source software (OSS) is a subject of academic studies in psychology, sociology, economics, or software engineering, including quantitative surveys. Developers' motivation [2], organization [3] and profiles [4] are investigated; Quantitative studies exist about code quality in OSS [5][6][7][8] and its dual, static analysis to uncover bugs [9][10]. Database surveys are available about market shares, or server exposure security issues [11]. This study is the first survey on the quality of relational database schemas in OSS. It provides a unique insight in the data engineering practice of these projects.

Codd's relational model [12] is an extension of the set theory to relations (tables) with attributes (columns) in which tuple elements are stored (rows). Elements are identified by keys, which can be used by tuples to reference one another between relations. The relational model is sound, as all questions (in the model) have corresponding practical answers and *vice versa*: the tuple relational calculus describes questions, and the mathematically equivalent relational algebra provides their answers. It is efficiently implemented by many commercial and open-source software such as Oracle, DB2 or SQLite. The *Structured Query Language* (SQL) is available with most relational database systems, although the detailed syntax often differs. The standardization effort also includes the *information schema* [13], which provides meta data about the schemas of databases through relations.

The underlying assumption of our study is that applications store precious transactional user data, thus should be kept consistent, non redundant, and easy to understand. We think that database features such as key declarations, referential integrity and transaction support help achieve these goals. In order to evaluate the use of database features in open-source software, and to detect possible design or implementation errors, we have developed a tool to analyze automatically the database structure of an application by querying its *information schema* and generating a report, and we have applied it to 407 open-source projects. Following MacCabe's metric to measure program complexities [14], several metrics address data models [15][16] or database schemata either in the relational [17][18] or object relational [19] models. These metrics rely on information not necessarily available from the database concrete schemas. We have rather followed the dual and pragmatic approach [20], which is not to try to do an absolute and definite measure of the schema, but rather to uncover issues based on static analyses. Thus the measure is relative to the analyses performed and results change when more are added.

Section II presents the methodology used in this study. We describe our tool, our grading strategy and the statistical validation used on the assertions derived from our analyses. Section III lists the projects by category and technology, and discusses similarities and differences depending on whether they run on MySQL or PostgreSQL. Section IV describes the results of our survey. The overall quality of projects is quite poor, as very few database schemas do not raise error-rated advices. Section V gives our conclusive thoughts.

II. METHODOLOGY

Our *Salix* automatic analyzer is based on the *information schema*. We discuss the queries, then describe the available advices, before presenting the statistical validation used.

A. Information schema queries

Our analyses are performed automatically by SQL queries on the databases meta data using the standard *information schema*. This relational schema stores information about the databases structure, including catalogs, schemas, tables, attributes, types, constraints, roles, permissions... The set of SQL queries used for this study are released as the *Salix* free software. It is based on *pg-advisor* [21], a PostgreSQL-specific proof of concept prototype developed in 2004. Some checks are inspired by [22][23][24] or similar to [25] others. Our tool creates a specific table for every advice by querying the *information schema*, and then aggregates the results in summary tables. It is fully relational in its conception: there is no programming other than SQL queries. The development of *Salix* uncovered multiple issues with both implementations of the *information schema*.

B. Advice classification and project grading

The 47 issues derived by our SQL queries on the standard *information schema* are named **advices**, as the user is free to ignore them. Although the performed checks are basic and syntactic, we think that they reflect the quality of the schemas. Each advice has a category (19 design, 13 style, 6 consistency, 4 version, 5 system), a severity (7 errors, 21 warnings, 14 notices, 5 informations), and a level (1 raised per database, 10 per schema, 27 per relation, 7 per attribute, 2 per role). The severity classification is arbitrary and must be evaluated critically by the recipient: most of them should be dealt with, but in some cases they may be justifiable. Moreover, detected errors do not imply that the application is not functional.

The 19 **design** advices focus on detecting design errors. Obviously, semantic error, say an attribute is in the wrong relation, cannot be guessed without understanding the application and thus are out of reach of our automatic analysis. We rather focus on primary and foreign key declarations, or warn if they are missing. The rate of non-null attributes is also checked, with the underlying assumption from our experience that most data are mandatory in a relation. We also check the number of attributes so as to detect a possible insufficient conception effort.

The 13 **style** advices focus on relation and attribute names. Whether a name is significant in the context cannot be checked, so we simply look at their length. Short names are discouraged as they would rather be used as aliases in queries, with the exception of *id* and *pk*. We also check that the same name does not represent differently typed data, to avoid confusing the user.

The 6 **consistency** advices checks for type and schema consistency in a project, such as type mismatches between a foreign key and the referenced key. As databases may also implements some of these checks, it is possible that some cases cannot arise.

The 4 **version** advices focus on database-specific checks, such as capabilities and transaction support, as well as homogeneous choices of back-end engines in a project. This category could also check the actual version of a database used looking for known bugs or obsolescence. Only MySQL-specific version advices are currently implemented.

Finally, the 5 **system** advices, some of which PostgreSQL-specific, check for weak passwords, and key and index issues.

These advices aim at helping the schema developer to improve its relational design. We also use them in our survey to grade projects with a mark from 0 to 10, by removing points each time an advice is raised, taking more points if the severity is high. The grading process is normalized using the number of possible occurrences, so that larger projects do not receive lower marks just because of the likelihood of having more issues for their size. Also, points are not removed twice for the same issue: for instance, if a project does not have a single foreign key, the same issue will not be raised again on every tables. Advices not relevant to our open-source database schema survey, e.g., weak password checks, were deactivated.

C. Survey statistical validation

The data collected suggest the influence of some parameters on others. These results deal with general facts about the projects (say foreign keys are more often used with PostgreSQL) or about their grading (say MySQL projects get lower marks). In order to determine significant influences, we applied Pearson's chi-square tests to compute probabilistic degrees of certainty. Each checked assertion is labeled with an expression indicating the degree of certainty of the influence of one parameter on another:

very sure The probability is 1% or less to get a result as or more remote from the average. Thus we conclude that there is an influence, with a very high degree of certainty.

rather sure The probability of getting such a result is between 1% and 5% (the usual statistical threshold). Thus there is an influence, with a high degree of certainty.

marginally sure The probability is between 5% and 25%: Such a result may have been obtained even if there is no influence. The statement must be taken with a pinch of salt.

not sure The probability is over 25%, or there is not enough available data to compute it. The test cannot asserts that there is a significant influence.

The rational for choosing Pearson's chi-square test is that it does not make any assumption about the distribution of values. However, it is crude, and possibly interesting and somehow true results may not be validated. Moreover, the test requires a minimal population, which is not easily reached on our small data set especially when criteria are crossed. Finally, it needs to define distinct populations: for grades or sizes, these populations are cut at the median value in order to perform the test on balanced partitions.

We also computed a correlation matrix to look for possible inter-parameter influence. The result suggested that the parameters are pretty independent beyond the obvious links (say the use of a non-transactional back-end is correlated with isolated tables), and did no help uncover significant new facts.

Category	Total	%	My	%	Pg	%	both	%	tabs	atts
CMS	70	17.2	61	20.2	1	3.7	8	10.3	39.2	6.6
System	36	8.8	17	5.6	1	3.7	18	23.1	28.1	11.9
Blog	24	5.9	20	6.6	0	0.0	4	5.1	28.2	7.1
Market	21	5.2	20	6.6	0	0.0	1	1.3	55.0	7.6
Project	20	4.9	11	3.6	3	11.1	6	7.7	29.7	7.2
Forum	19	4.7	17	5.6	0	0.0	2	2.6	23.1	8.3
Accounting	16	3.9	9	3.0	6	22.2	1	1.3	93.1	8.4

TABLE I
MAIN CATEGORIES OF PROJECTS, WITH COUNTS, DATABASE SUPPORT AND SIZES

Technology	Total	%	My	%	Pg	%	both	%	tabs	atts
PHP	311	76.4	262	86.8	7	25.9	42	53.8	32.2	7.3
C	34	8.4	9	3.0	5	18.5	20	25.6	22.4	11.7
Java	18	4.4	7	2.3	5	18.5	6	7.7	57.3	9.4
Perl	18	4.4	9	3.0	4	14.8	5	6.4	50.5	7.1

TABLE II
MAIN TECHNOLOGIES OF PROJECTS, WITH COUNTS, DATABASE SUPPORT AND SIZES

III. PROJECTS

We discuss the projects considered in this study, grouped by categories, technologies, sizes and release dates. We first present how projects were selected, and then an overview.

A. Project selection

We have downloaded 407 open-source projects starting in the first semester of 2008, adding to our comparison about every project that uses either MySQL or PostgreSQL that we could find and install with reasonable time and effort. The database schemas included in this study are derived from a dump of the database after installation, or from the creation statements when found in the sources. These projects were discovered from various sources: lists and comparisons of software on Wikipedia and other sites; package dependencies from Linux distributions requiring databases; security advisories mentioning SQL; searches on SourceForge.

Some projects were fixed manually because of various issues, such as: the handling of double-dash comments by MySQL, attribute names (*e.g.*, `out`) rejected by MySQL, bad foreign key declarations or other incompatibilities detected when the projects were forced to use the InnoDB back-end instead of MyISAM, or even some PostgreSQL table definitions including a MySQL specific syntax that were clearly never tested. A particular pitfall of PostgreSQL is that by default syntax errors in statements from an SQL script are ignored and the interpreter simply jumps to the next statement. When installing a project, the flow of warnings often hides these errors. Turning off this feature requires modifying the script, as no command option disables it. More than a dozen PostgreSQL projects contained this kind of issues, which resulted in missing tables or ignored constraint declarations.

B. Overview of projects

We have studied the relational schemas of 407 (see [26] for the full list) open-source projects based on databases: 380 of these run with MySQL, 105 with PostgreSQL, including 78 on both. A project supporting PostgreSQL is very likely

to support also MySQL (74%), although the reverse is not true (only 20%) (*very sure*), outlining the relative popularity of these tools. Only 27 projects are PostgreSQL specific. Although there is no deliberate bias in the selection process described in the previous section, where we aimed at completeness, some implicit bias remains nevertheless: for instance, as we can speak mostly English and French, we found mostly international projects advertised in these tongues; Table I shows main project categories, from the personal mundane (game, homepage) to the professional serious (health-care, accounting, system). Table II shows the same for project technologies. Projects in rare categories or using rare technologies do not appear in these cut-off tables. The result is heavily slanted towards PHP web applications (76%), which seems to reflect the current trend of open-source programming as far as the number of projects is concerned, without indication of popularity or quality. The ratio of PHP projects increases from PostgreSQL only support (25%) to both database support (53%) (*rather sure*) to MySQL only support (86%) (*very sure*): PHP users tend to choose specifically MySQL.

The survey covers 16104 tables (MySQL 11303, PostgreSQL 4801) containing 139092 attributes (MySQL 93960, PostgreSQL 45132). The project sizes in tables average at 33.2, median 17 (from 1 to 607 tables), with 2 to 10979 attributes. MySQL projects average at 30 tables, median 16 (from 1 to 466), with 247 attributes (from 2 to 9725), while PostgreSQL projects average 46 tables, median 20 (from 1 to 607), with 430 attributes (from 6 to 10979 attributes). The largest MySQL project is OSCARMMASTER, and the largest PostgreSQL project is ADEMPIERE. Detailed table counts raise from projects with MySQL only support (average 28.3, median 16), to both databases (average 34.5, median 18) or PostgreSQL only (average 81.1, median 31). MySQL-only projects are smaller than other projects (*marginally sure*): more ambitious projects seem to use feature-full but maybe less easy to administrate PostgreSQL. However obvious this assertion would seem, the statistical validation is weak because of the small number of projects with PostgreSQL. MySQL projects that use the InnoDB back-end are much larger than

Advice	Lvl.	Cat.	Sev.	MySQL			PostgreSQL		
				Proj	%	Adv	%	Proj	%
Schema without any FK	sch.	design	error	339	89	339	89	58	55
Tables without PK nor Unique	table	design	error	218	57	1272	11	67	63
FK type mismatch	table	consist.	error	2	0	17	0	10	9
Backend engine inconsistency	sch.	version	error	26	6	26	6	0	0
FK length mismatch	table	consist.	error	3	0	5	0	1	0
Integer PK but no other key	table	design	warn	345	90	6119	54	89	84
Homonymous heterogeneous attributes	att.	style	warn	242	63	1998	2	66	62
Unsafe backend engine used in schema	sch.	version	warn	338	88	338	88	0	0
Isolated Tables	table	design	warn	25	6	857	7	35	33
Tables without PK but with Unique	table	design	warn	98	25	341	3	15	14
Unique nullable attributes	att.	design	warn	58	15	226	0	22	20
Nullable attribute rate over 80%	sch.	design	warn	24	6	24	6	21	20
Redundant indexes	table	system	warn	0	0	0	0	22	20
Attribute name length too short	att.	style	warn	22	5	65	0	15	14
Large PK referenced by a FK	table	design	warn	9	2	99	0	15	14
Composite Foreign Key	table	design	warn	5	1	19	0	8	7
Table name length too short	table	style	warn	9	2	10	0	6	5
FK not referencing a PK	table	design	warn	1	0	12	0	7	6
Redundant FK	table	system	warn	1	0	1	0	2	1
Non-integer Primary Key	table	design	note	214	56	1950	17	66	62
Attribute count per table over 20	table	design	note	188	49	535	4	54	51
MySQL is used	base	version	note	380	100	380	100	0	0
Tables with Composite PK	table	design	note	164	43	1583	14	54	51
Attribute name length quite short	att.	style	note	159	41	609	0	42	40
Attribute named after its table	att.	style	note	100	26	1473	1	35	33
Table without index	table	system	note	0	0	0	0	53	50
Nullable attribute rate in 50-80%	sch.	design	note	62	16	62	16	24	22
Table name length quite short	table	style	note	56	14	81	0	24	22
Table with a single attribute	table	design	note	59	15	360	3	22	20
Mixed attribute name styles	table	style	note	89	23	752	6	1	0
Mixed table name styles	sch.	style	note	31	8	100	26	3	2
Attribute name length short	att.	style	info	267	70	2240	2	69	65
Unsafe backend engine used on table	table	version	info	338	88	8746	77	0	0
Nullable attribute rate in 20-50%	sch.	design	info	107	28	107	28	39	37
Table name length short	table	style	info	99	26	197	1	32	30

TABLE III
LIST OF RAISED ADVICES AND DETAILED COUNTS ABOUT THE 407 PROJECTS

their MyISAM counterpart (*very sure*) and are comparable to projects based on PostgreSQL, with 58 tables on average. The number of attributes per table is comparable although smaller for MySQL (average 8.3 – median 7.0) with respect to PostgreSQL (average 9.4 – median 7.0).

The per-category tables (*tabs*) and attributes-per-table (*atts*) counts shows that *accounting*, *health-care* and *market* projects are more ambitious than other categories (*marginally sure*). The per-technology analysis counts suggests that *Perl*, *Python* and *Java* projects are larger than those based on other technologies (*rather sure*).

These projects are mostly recent, taking their status at an arbitrary common reference date chosen as March 31, 2009: 257 (63%) were updated in the last year, including 141 (34%) in the last six months, and the others are either obsolete or very stable. The rate of recent projects raises from MySQL-only projects (57%) to projects with both support (79%) (*very sure*) or with PostgreSQL support at (81%) (*very sure*). However there is no significant difference on the recent maintenance figure between projects that are PostgreSQL-only and projects with both databases support. Projects that include PostgreSQL support were updated more recently than those

based on MySQL only.

IV. SURVEY RESULTS

We now analyze the open-source projects of our survey by commenting actual results on MySQL and PostgreSQL, before comparing them. Table III summarizes the advices raised for MySQL and PostgreSQL applications. The first four columns give the advice title, level, category and severity. Then four columns for each database list the results. The first two columns hold the number of projects (*i.e.* schema) tagged and the overall rate. The last two columns give the actual number of advices and rate, which varies depending on the level. A per-project aggregate is also available online [26].

A. Primary keys

A majority of MySQL projects (218 – 57%) have at least one table without neither a primary key nor a unique constraint, and this is even worse with PostgreSQL projects (67 – 63%). The certainty of the observation (*marginally sure*) on MySQL-only vs PostgreSQL-only is low because of the small number of projects using the later. As 11% of all MySQL tables and 19% of all PostgreSQL tables do not have any

key, the view of relations as sets is hindered as tuples are not identified, and data may be replicated without noticing.

A further analysis gives some more insight. For MySQL, 42% of tables without key do have some KEY option for indexes, but without the UNIQUE or PRIMARY keyword that makes it a key. Having KEY not always declaring a key was clearly a bad design choice. A little 2% of tables without key have an *auto increment* attribute, which suggest uniqueness in practice, but is not enforced. Also, the missing key declaration often seems to be composite. Some tables without key declarations are intended as one tuple only, say to check for the version of the schema or configuration of the application. Similarly, 28% of PostgreSQL tables without key have an index declared. Moreover, 19% have a SERIAL (auto incremented) attribute: Many designers seem to assume wrongly that SERIAL implies a key. A comment found in the SQLGREY project source suggests that some keys are not declared because of MySQL key size limits.

A simple integer primary key is provided on 60% of tables, with a significantly decreasing rate from MySQL-only (65%) to both database support (59%) (*very sure*) down to PostgreSQL-only support (39%) (*very sure*). If these primary keys were non-semantic numbers to identify tuples, one would expect at least one other key declared on each table to identify the underlying semantic key. However it is not the case: most (84%) of these tables do not have any other key. When a non simple primary key is available, it is either based on another type or a composite key. The composite keys are hardly referenced, but as the foreign keys are rarely declared one cannot be sure, as shown in the next section.

B. Referential integrity

Foreign keys are important for ensuring the data consistency in a relational database. They are supported by PostgreSQL, and by MySQL but with some back-end engines only. In particular, the default MyISAM back-end does not support foreign keys, and this feature was deemed noxious in previous documentations: Version 3.23 includes a *Reasons NOT to Use Foreign Keys constraints* Section arguing that they are only useful to display diagrams, hard to implement and terrible for performance. Foreign key constraints are introduced with the InnoDB engine starting with *MySQL 3.23.44* in January 2001. Although the constraints are ignored by the default MyISAM engine, the syntax is parsed, and triggers the creation of indexes. Version 5.1 documentation has a *Foreign Keys* Section praising the feature, as it *offers benefits*, although it slows down the application. Caveats describe the inconsistencies that may result from *not* using transactions and referential integrity. From a pedagogical perspective, this is a progress.

Foreign key constraints have long been a missing or avoided feature in MySQL and this seems to have retained momentum in many projects, as it is not supported by the default engine: few MySQL projects (41 – 10% of all projects, 60% of those with InnoDB) use foreign key constraints. The foreign key usage rate is significantly higher (22%) when considering projects supporting both databases (*marginally sure*).

Among MySQL projects, 312 (82%) use only the default MyISAM back-end engine, thus do not have any foreign key

checks enabled. In the remainder, 42 (11%) use only InnoDB, and 26 (6%) use a combination of both. More projects (20 – 25%) rely on InnoDB among those supporting both MySQL and PostgreSQL (*rather sure*). A third of InnoDB projects (26 – 38%) are not consistent in their engine choice: 35% of tables use MyISAM among the 68 InnoDB projects. A legitimate reason for using MyISAM tables in an InnoDB project is that FULLTEXT indexes are only available with the former engine. However, this only applies to 11 tables in 6 projects, all other 1403 MyISAM tables in InnoDB projects are not justified by this argument. A project may decide to store transient data in an unsafe engine (*e.g.*, memory) for performance reason and possibly without any risk of losing data, but this optimization is beyond our tool and is reported as an error. This case is rare, as it represents only 13 tables in 6 projects. About 26% of tables use MyISAM as a default implicit choice in InnoDB projects, similar to 26% when considering all MySQL projects. Some engine inconsistencies seems due to forgotten declarations falling back to the default MyISAM engine.

We have forced the InnoDB back-end engine for all MySQL projects: 22 additional projects declare 92 new foreign key constraints previously ignored. These new foreign keys are very partial, targeting only some tables. They allow to uncover about two dozen issues, either because the foreign key declaration were failing (say from type errors detected by MySQL) or thanks to analyses from our tool. Additional checks based on foreign keys cannot be raised on schemas that do not declare any of them. Thus *isolated tables* warnings must be compared to the number of projects that do use referential constraints: 25 – 60% of these seem to have forgotten at least some foreign keys, and it is actually the case by checking some of these projects manually.

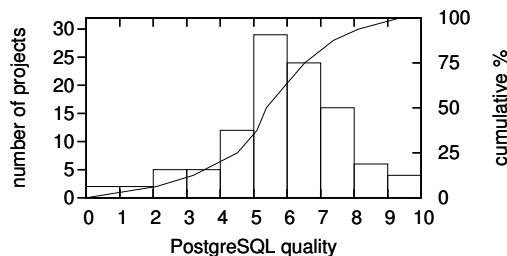
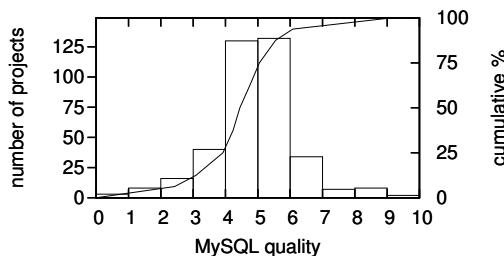
The foreign key usage is better with PostgreSQL projects, although it is still a minority (47 projects – 44%). This rate is close to the foreign key usage of MySQL projects when considering InnoDB projects only. It gives a better opportunity for additional advices to be checked. The foreign key usage rate raises significantly to 78% when considering PostgreSQL-only projects vs dual support projects (*very sure*).

On the very few projects with partial foreign key declarations, several of these declaration reveal latent bugs, including type mismatch, typically CHAR targeting a VARCHAR or vice-versa, or different integers, and type length mismatch, usually non matching VARCHAR sizes. There are 22 such bugs found out of the small 1387 declared MySQL attribute constraints, and 156 among the 3861 PostgreSQL constraints. There are also 130 important warnings related to foreign keys raised for MySQL, and 227 for PostgreSQL. If this ratio of errors is projected on a the number of tables involved, hundreds additional latent bugs could be detected if the developers were to declare the referential constraints.

C. Miscellaneous issues

More issues were found about style, attribute constraints and by comparing projects with dual database support.

There is 10679 noticeable style issues raised from our analyses (5088 for MySQL, 5591 for PostgreSQL), relating

TABLE IV
QUALITY PER DECILE

Size	nb	MySQL projects				
		avg	σ	min	med	max
small	134	4.6 ± 1.3	0.5	4.5	9.1	
medium	132	4.3 ± 1.2	0.0	4.4	8.7	
large	114	4.4 ± 1.3	0.0	4.5	8.2	

Size	nb	PostgreSQL projects				
		avg	σ	min	med	max
small	37	5.1 ± 1.9	0.0	5.2	9.4	
medium	27	5.7 ± 1.6	2.0	5.3	9.3	
large	41	5.4 ± 2.0	0.0	5.7	9.1	

TABLE V
QUALITY PER SIZE

Category	nb	MySQL projects				
		avg	σ	min	med	max
project	17	4.5 ± 1.1	1.4	4.8	6.2	
system	35	4.4 ± 1.4	0.0	4.7	7.0	
blog	24	4.5 ± 0.9	2.5	4.5	7.2	
forum	19	4.3 ± 0.9	2.4	4.4	5.7	
cms	69	4.3 ± 0.9	0.5	4.3	6.3	
market	21	4.0 ± 1.4	1.9	4.3	8.2	

Category	nb	PostgreSQL projects				
		avg	σ	min	med	max
accounting	7	6.1 ± 2.3	2.0	6.5	9.1	
cms	9	6.4 ± 1.3	4.1	6.2	8.1	
irc	7	5.4 ± 1.7	2.0	5.7	7.4	
project	9	5.8 ± 1.5	4.4	5.3	9.3	
system	19	4.9 ± 1.7	2.0	5.0	9.0	
mail	8	4.9 ± 1.6	3.0	4.8	7.5	

TABLE VI
QUALITY PER PROJECT MAIN CATEGORIES

Techno.	nb	MySQL projects				
		avg	σ	min	med	max
java	13	4.7 ± 2.4	0.0	5.2	7.8	
c	29	4.7 ± 1.5	2.0	4.7	8.4	
perl	14	4.1 ± 2.1	0.5	4.6	8.7	
php	304	4.4 ± 1.1	0.0	4.4	8.4	

Techno.	nb	PostgreSQL projects				
		avg	σ	min	med	max
java	11	6.0 ± 2.6	0.0	6.5	9.3	
perl	9	5.8 ± 1.8	2.0	6.1	8.1	
php	49	5.2 ± 1.7	0.0	5.4	8.2	
c	25	5.0 ± 1.7	2.0	5.1	9.0	

TABLE VII
QUALITY PER PROJECT MAIN TECHNOLOGIES

Date	nb	MySQL projects				
		avg	σ	min	med	max
recent	125	4.4 ± 1.2	1.3	4.4	8.4	
older	255	4.4 ± 1.3	0.0	4.4	9.1	

Date	nb	PostgreSQL projects				
		avg	σ	min	med	max
recent	52	5.3 ± 1.6	0.0	5.3	9.3	
older	53	5.5 ± 2.0	0.0	5.7	9.4	

TABLE VIII
QUALITY PER PROJECT RELEASE DATE

to table or attribute names, including a number of one-letter attribute names or two-letters table names. The *id* attribute name is used in the SLASH project with up to 6 different types, mixing various integers and fixed or variable length text types. In PHPETITION, a *date* attribute has types DATE, DATETIME or VARCHAR. 80% of MySQL projects and 79% of PostgreSQL have such style issues.

Many projects does not bother with NOT NULL attribute declarations: 86 MySQL projects (22%) and 45 PostgreSQL projects (42%) have over half of their attributes null-able. This does not reflect the overall use of constraints: for MySQL, the average number of key-related constraints per table is 1.06 (from KANNEL 0.00 to OPENMRS 3.54), while for PostgreSQL it is 1.20 (from ANDROMEDA 0.00 to ADEMPIERE 4.25). Project ANDROMEDA is astonishing: there is not a single constraint declared (no primary key, no foreign key, no unique, no not

null) on the 180 tables, although there are a number of non-unique indexes and of sequences.

It is interesting to compare the schemas of the 78 projects available with both databases. This dual support must not be taken at face value: PostgreSQL support is often an afterthought and is not necessarily functional, including project such as ELGG, TAGADASH, QUICKTEAM or TIKIWIKI where some PostgreSQL table declarations use an incompatible MySQL syntax; 27 (34%) projects have missing tables or attributes between the MySQL and PostgreSQL versions: 190 tables and 173 individual attributes are missing or misspelled one side or another. Among the missing tables, 73 look like some kind of sequence, and thus might be possibly legitimate, although why the *auto increment* feature was not satisfactory is unclear. At the minimum, the functionalities are not the same between MySQL and PostgreSQL versions for those projects.

D. Overall quality

We have computed a synthetic project quality evaluation ranging from 10 (good) to 0 (bad) by removing points based on advice severity (error, warning, notice), level (schema, table, attribute) and project size. The MySQL projects quality average is 4.4 ± 1.3 (from 9.1 GENOVAWEB to 0.0 OSCARMASTER), significantly lower than PostgreSQL 5.4 ± 1.8 (from 9.4 COMICS to 0.0 NURPAWIKI) (*very sure*). This does not come as a surprise: most MySQL projects choose the default data-unsafe MyISAM engine, hence incur a penalty. Also, the multiplicity of MySQL back-ends allows the user to mix them unintentionally, what is not possible with PostgreSQL. When all MySQL-specific advices are removed, the quality measure is about the same for both databases. However, as PostgreSQL schemas provide more information about referential integrity constraints, they are also penalized as more advices can be raised based on the provided additional information.

Table IV shows the projects per quality decile. The PostgreSQL project quality is more spread than MySQL projects (*marginally sure*). Table V compares the quality of projects according to size, where small is up to 9 tables, medium up to 29, and large otherwise. The quality is quite evenly distributed among sizes, which suggests that our effort to devise a size-neutral grading succeeded. Table VI compares quality based on the project categories. The number of projects in each category is too small to draw deep conclusions. Table VII addresses the technology used in the project: Java leads while PHP is near bottom. PHP projects take less care of their relational design (*very sure*). Finally, Table VIII shows that quality evaluation is basically the same for recent and older projects.

V. CONCLUSION

This is the first survey on the quality of relational schemas in open-source software. The overall quality results are worse than envisioned at the beginning of the study. Although we did not expect a lot of perfect projects, having so few key declarations and referential integrity constraints came as a surprise. We must acknowledge that our assumption that data are precious, and that the database should help preserve its consistency by enforcing integrity constraints and implementing transactions, is not shared by most open-source projects, especially when based on MySQL and PHP. This is illustrated by bug report 15441 about missing keys on tables in MEDIAWIKI: it had no visible effect after two years.

The first author contributed both to the best PostgreSQL project (COMICS), and to one of the worst MySQL project (SLXBBL), which is *Salix* executed on its own schema! This deserves an explanation: COMICS is a small database used for teaching SQL. The normalized schema emphasizes clarity and cleanliness with a pedagogic goal in mind. Even so, the two raised warnings deserve to be fixed, although one would require an additional attribute. SLXBBL tables generate a lot of errors, because they are views materialized for performance issues. Also, they rely on MyISAM because some SQL create table statements must be compatible with both MySQL and PostgreSQL to ease the tool portability. Nevertheless, the comparison of schemas allowed to find one bug: an attribute had a different name, possibly because of a bad copy-paste.

We have released our *Salix* tool as a free software. As it is fully relational and relies on standards, it may be installed in any compliant database to help improve schemas.

Acknowledgement – Thanks to Pierre Jouvelot.

REFERENCES

- [1] A. Deshpande and D. Riehle, “The Total Growth of Open Source,” in *4th Conference on Open Source Systems (OSS)*. Springer Verlag, 2008, pp. 197–209.
- [2] A. Hars, “Working for free? motivations for participating in open-source projects,” *International Journal of Electronic Commerce*, vol. 6, pp. 25–39, 2002, also IEEE 34th Hawaii International Conference on System Sciences 2001.
- [3] K. Crowston and H. Annabi, “Effective work practices for software engineering: Free/libre open source software development,” in *in Proc. of WISER*. ACM Press, 2004, pp. 18–26.
- [4] D. M. Nichols and M. B. Twidale, “The usability of open source software,” *First Monday*, vol. 8, 2003.
- [5] B. Mishra, A. Prasad, and S. Raghunathan, “Quality and Profits Under Open Source Versus Closed Source,” in *ICIS*, no. 32, 2002.
- [6] I. Stamelos, L. Angelis, A. Oikonomou, and G. L. Bleris, “Code quality analysis in open-source software development,” *Information Systems Journal, 2nd Special Issue on Open-Source*, vol. 12, no. 1, pp. 43–60, Feb. 2002, blackwell Science.
- [7] E. Capra, C. Francalanci, and F. Merlo, “An Empirical Study on the Relationship among Software Design Quality, Development Effort and Governance in Open Source Projects,” *IEEE Software Engineering*, vol. 34, no. 6, pp. 765–782, nov-dec 2008.
- [8] R. Gobelle, “The FOSSology Project,” in *Working Conference on Mining Software Repositories*, no. 5, Leipzig, Germany, May 2008.
- [9] Coverty, “Coverty scan open source report,” Coverty, White Paper, 2009.
- [10] Veracode, Inc, “State of security report.” White paper, Mar. 2010.
- [11] D. Litchfield, “The Database Exposure Survey 2007,” NGSSoftware Insight Security Research (NISR), Nov. 2007.
- [12] E. F. Codd, “A relational model for large shared databanks,” *Communications of the ACM*, vol. 13, no. 6, pp. 377–387, Jun. 1970.
- [13] ISO/IEC, Ed., *9075-11:2003: Information and Definition Schemas (SQL/Schemata)*. ISO/IEC, 2003.
- [14] T. J. MacCabe, “A Complexity Measure,” *IEEE Software Engineering*, vol. SE-2, no. 4, pp. 308–320, Dec. 1976.
- [15] M. Piattini, M. Genero, C. Calero, and G. Alarcos, “Data model metrics,” in *In Handbook of Software Engineering and Knowledge Engineering: Emerging Technologies*, World Scientific, 2002.
- [16] M. Genero, “A survey of Metrics for UML Class Diagrams,” *Journal of Object Technology*, vol. 4, pp. 59–92, Nov. 2005.
- [17] H. M. Sneed and O. Foshag, “Measuring legacy database structures,” in *European Software Measurement Conference (FESMA'98)*, Hooft and Peeters, Eds., 1998.
- [18] M. Piattini, C. Calero, and M. Genero, “Table Oriented Metrics for Relational Databases,” *Software Quality Journal*, vol. 9, no. 2, pp. 79–97, 2001.
- [19] A. L. Baroni, C. Calero, F. Ruiz, and F. Brito e Abreu, “Formalizing object-relational structural metrics,” in *Conference of APSI, Lisbon*, no. 5, Nov. 2004.
- [20] A. Bessey, K. Block, B. Chelf, A. Chou, B. Fulton, S. Hallem, C. Henri-Gros, A. Kamsky, S. McPeak, and D. Engler, “A Few Billion Lines of Code Later: Using Static Analysis to Find Bugs in the Real World,” *Communication of the ACM*, vol. 53, no. 2, pp. 66–75, Feb. 2010.
- [21] F. Coelho, “PG-Advisor: proof of concept SQL script,” Mailed to pgsql-hackers, Mar. 2004.
- [22] J. Currier, “SchemaSpy: Graphical database schema metadata browser,” Source Forge, Aug. 2005.
- [23] B. Schwartz and D. Nichter, “Maatkit,” Google Code, 2007, see *duplicate-key-checker* and *schema-advisor*.
- [24] J. Berkus, “Ten ways to wreck your database,” O'Reilly Webcast, Jul. 2009.
- [25] A. M. Boehm, M. Wetzka, A. Sickmann, and D. Seipel, “A Tool for Analyzing and Tuning Relational Database Applications: SQL Query Analyzer and Schema EnHancer (SQUASH),” in *Workshop über Grundlagen von Datenbanken*, Jun. 2006, pp. 45–49.
- [26] F. Coelho, “Database quality survey projects and results,” Nov. 2010, detailed list of projects considered in *A Field Analysis of Relational Database Schemas in Open Source Software*, report A/423/CRI. [Online]. Available: <http://www.coelho.net/salix/projects.html>

The Use of Data Cleansing in Mobile Devices

María del Pilar Angeles, Francisco García-Ugalde
 Facultad de Ingeniería
 Universidad Nacional Autónoma de México
 México, D.F.
pilarang@unam.mx, fgarciau@servidor.unam.mx

Abstract—People receive information on PDA, mobile phone or mp3 player. If such information was correct, current, useful and usable, users would be able to use it immediately from any mobile device with no requirement of a personal computer to assess, clean and integrate data. The present research proposes the utilization of a data cleansing framework for the improvement of data quality in mobile devices.

Keywords-data quality;mobile devices; data cleansing.

I. INTRODUCTION

Nowadays, people utilize huge amount of data coming from a range of mobile devices under different data storage and data formats in order to make business. It is very well known that companies having useful information have better possibilities to exploited it, and make better informed decisions. Companies establishing better strategies are able to become leaders and business competitive. For instance, employees are more productive by keeping in touch through text messages to customers or colleagues while they are attending a meeting out of the office.

However, since information come from a number of data sources, such data sources are structured, unstructured or semi structured and the process of information integration is not a trivial task, due to semantic and syntactic heterogeneities degrading data quality as a consequence.

Typical causes of poor data quality are data entry errors, wrong or unspecific metadata, lack of enforcement or not defined appropriately integrity constraints [1], [2]. People are used to edit information from the songs they legally downloaded from the internet, or type personal information as business contacts within mobile phones o personal digital assistants (PDA) in order to make phone calls, send text messages or emails to arrange a meeting for business. For instance, in the case of an mp3 player, hundreds or thousands of songs can be stored (on IDv3 tags) under different genres, albums, singers, etc. The management of information becomes chaotic, tiring and annoying. Unfortunately, having duplicated or obsolete personal details stored in an agenda is not unusual and could be the cause of losing business opportunities. The problem increases after a software upgrade or data migration to other mobile phone.

There are a number of software tools for the edition of audio files tags, or for the management of contacts in mobile phones or electronic organizers namely Personal Information Management (PIM). However, such tools are not very practical when there are hundreds of rows to manage. Users have to select by hand all the records they want to update.

David Alcudia-Aguilera
 Facultad de Ciencias
 Universidad Juárez Autónoma de Tabasco
 Tabasco, México
072H3024@alumno.ujat.mx

Furthermore, if there are 200 songs of the same genre, but this field has been captured in 20 different forms, the genres are semantically equal but syntactically different, for instance, ROCK, rock, RoCK, etc. The readability and usability of the mp3 player is affected.

Data quality patterns and data matching have been developed recently for the detection and correction of data errors within the process of data cleansing [6], [7]. Data cleansing has being widely used on data warehousing [3], [4], [5], but not on data stored in mobile devices such as mobile phones, electronic organizers o audio players.

The following section is aimed to explain the problem of poor data quality during integration process within mobile devices. The third section provides the bases of context of Data Quality, data cleansing and data quality patterns. The fourth section presents a Data Cleansing Framework for mobile devices. The fifth section concludes with the main findings and implications for future research.

II. POOR DATA QUALITY WITHIN MOBILE DEVICES

A. Poor data quality on mp3 files

In this section, we analyze the problem of data cleansing in mp3 files. It is common to have a large list of mp3 files in a directory with several files wrong documented, repeated or incomplete.

- Problem description

“I upgraded my mp3 player to the current version, and I am enjoying the new features. However, my album folders are not sorted properly (even though they appear as intended in the mp3 organizer on my PC).

Some songs are not with the album they are part of. Other albums are not where they should be chronologically, numerically, or alphabetically.”

An ID3 tag is a data container within an MP3 audio file stored in a prescribed format. This data commonly contains the Artist name, Song title, Year and Genre of the current audio file. However, even if the music titles were rightly spelled and correct, if they were longer than 17 characters, users are unable to identify which track will be played because display restrictions of some mp3 players that only 16 characters would be displayed.

Large repositories of information are frequently incomplete, redundant or corrupted. The same problem is arising within small data sets. For instance, is common to have a music folder in our computer containing same song several times. In addition, is usual to have non descriptive or invalid filenames.

Trying to resolve this problem “by hand” is time consuming as presented in the following section.

- Typical solution

MP3 organizers allow users a number of features such as localize songs performed by a main artist under one name, preferably with the correct spelling within one folder. Make sure album titles are the same and correctly-spelled throughout the album. Add the album artwork to the album if you wish. Find out the original albums of songs in "Greatest Hits" to avoid having two pointless albums. Put any soundtracks in the genre "Soundtrack" for easy access. Edit and check spelling other genre classifications (Rock, Pop, Indie). Organize music by time. Find duplicate songs. Select a song and get info. Type in the correct artist name and the already mentioned info, Edit multiple songs by the same artist, by clicking the first song, then hold down CTRL and click the others by that artist, right-click one of them and select get info. The mp3 organizer program will ask for confirmation to edit multiple songs information. The music organizer programs might offer a full range of features. However, if they are all manual, they are not practical approaches in the case of hundreds or thousands of contacts or music files.

B. Poor data quality within Smart phones or Personal Digital Assistants

- Problem Description

In the late 90's the PDA offered users the possibility to become mobile professionals enable to better manage information through a personal organizer on the go. Years later smart phones appear and the contacts information were migrated from one platform to other, as the software were not compatible, one contact with three phone numbers (home, job, mobile) on the PDA became three entries under the same contact name with one phone number each. Some entries were not identified containing the phone number only. Other problem was regarding ciphering, due to incompatibility between Pocket PC o Smartphone and mobile phones because management software does not support ciphering, making data corrupted or even lost.

- Typical Solution

Users spent hours trying to find which phone number belong to which person and if it was office number or mobile phone number. Therefore users definitely delete extra entries. Most people just delete the entries and keep only the most known contact numbers missing business opportunities. The migration process requires software installed on the personal computer. If the contact information was barely captured, the migration would perform with no prove. However, if the contact lists contains for instance '+' for international phone calls, such character would not be reflected in the new contact list and changes have to be performed by hand again.

There are open software that provides mobile synchronization and push email solutions to mobile phones. It includes a mobile server that pushes email to mobile phones from a number of mail servers. It enables users to synchronize contacts, calendar, tasks and notes. Offering a portal that lets anyone get email on their mobile phone and that syncs PIM data to their devices. Users are required to provide personal details such as email user and password and

their phone number. The main issue here is data security and user information confidentiality.

III. THE CONTEXT OF DATA QUALITY

The subjective nature of the term Data Quality (DQ) has allowed the existence of general definitions such as "fitness for use" in [18], which implies that quality depends on customer requirements.

The definition established by Redman [11], suggests that data quality can be obtained by comparing two data sources. "A datum or collection of data X is of higher or (better) quality than a datum or collection of data Y if X meets customer needs better than Y".

Recently, data quality has been defined as "the capability of data to be used effectively economically and rapidly to inform and evaluate decisions" [15]. Such definition considers data quality not as the end but the means for making informed decisions.

Data quality is characterized by quality criteria or dimensions such as accuracy, completeness, consistency, and timeliness in several approaches such as [1], [5], [16], [17], [18] mainly because classification facilitates the characterization and definition of an overall quality.

In the case of mobile applications, there is not massive information like in data warehouse environments but sufficiently relevant for making business. For instance, a mobile application developed to collect census data is detailed in [9].

The quality properties considered in this research are accuracy, amount of data, Format appropriateness, format precision, representation consistency, uniqueness, completeness, usability, usefulness, which will be explained in detail as follows:

Accuracy has been considered in [16] as the "measure of the degree of agreement between a data value and the source agreed to be correct".

The quality property amount of data refers to the extent to which the volume of data is appropriate for the task at hand.

Format appropriateness refers to the capability of a data format to be more appropriate than other because it is better suited to users' needs [11].

The Format precision refers to the set of symbolic representations are sufficiently precise to distinguish among elements in the domain that must be distinguished by the users, there are values correctly represented, values not represented (missing) and values that do not correspond with real world. The data cleansing process should support several character sets in order to be suitable for different languages.

Representation consistency refers to whether physical instances of data are in accord with their format; the constraints are posed in terms of membership in the set of a symbolic representation [11], or in terms of conformance to a format standard.

The Uniqueness property is the extent where an entity from the real world is represented once.

Usability is the extent to which data are used for the task at a hand with acceptable effort.

The Usefulness property refers to the degree where using data provides benefit on the performance on the job, in other words the extent to which the user believes data would be useful for the task at hand. The data cleansing process shall be effective enough to produce clean data and useful.

In order to correct, standardize and consequently, to improve data quality, data cleansing has emerged to define and determine error types, search and identify error instances, and correct the errors.

A. Data cleansing

“Data cleansing is applied especially when several databases are merged. Records referring to the same entity are represented in different formats in different data sets or are represented erroneously. Thus, duplicated records will appear in the merged database. This problem is known as merge/purge problem.” [3].

According to [14] the most common methods utilized for error detection are: statistical methods through standard deviation, quartile ranges, regression analysis, etc. [13], [12]; clustering is a data mining method to classify data in groups to identify discrepancies; pattern recognition based methods to identify records that do not fit into a certain specific pattern and association rules to find dependencies between values in a record [3].

Data cleansing is commonly performed in offline time, which is unacceptable for operational systems. Therefore, cleansing is often regarded as a pre-processing step for Knowledge Discovery in Databases and Data Mining systems during the Extraction Transformation and Load (ETL) process. However, it is still a very time consuming task, “The process of data cleansing is computationally expensive on very large data sets and thus it was almost impossible to do with old technology” [3]. The main objective within the present research is to use the data quality pattern recognition and record matching as useful data cleansing tools within mobile devices. The main advantage is the feasibility of using data cleansing in small quantity of data contained in mobile devices.

Previous works and approaches related to the subject of this paper try to resolve and propose methods for the issues of data cleansing, data quality, and resolution of data inconsistencies. Some of them are presented as follows:

Loshin [2], approached the problem of Data Quality. Their objectives were to develop an automated procedure to assess the quality of civil infrastructure monitoring data and to explore how effectively the data can be cleansed using the assessments results. Loshin proposed seven different cleansing techniques. (1) the do nothing technique leaves the data set as is, (2) the correction technique applies a constant additive or multiplicative factor to the data, (3) the replace technique replaces the original value with a new value; the new value might be an average over the attribute values or a value generated by a prediction algorithm, (4) the split/combine technique splits a single record into two records or combines two records into a single record, (5) the insertion technique duplicates data from one part of the data set and inserts the duplicate data into a part where data are missing, (6) the remove technique removes data from the

data set, and (7) Meta-data techniques add new information to the data set, in the form of confidence intervals, weights, or flags.

Maletic and Marcus [3] confront the problem of data cleansing and automatically identifying potential errors in data sets. They presented three phases in the process of data cleansing: (1) define and determine error types, (2) search and identify error instances, and (3) correct the uncovered errors.

In addition, Hernandez and Stolfo [4] addressed the problem of merging multiple databases of information about common entities (the Merge/Purge Problem). They developed a system for accomplishing this data cleansing task.

The next section presents the use of pattern recognition to identify records that do not fit into certain specific problem and correct them by data standardization as part of the data cleansing process.

B. Detection of Data Quality Patterns

The detection of data quality patterns aims to the improvement of data quality by ensuring data consistency. Once the data pattern is detected, the metadata is updated with the right definition of data format. For instance, the integrity constraint is updated or defined according to the data pattern. Therefore, the integrity constraint is forced into the data and data become consistent.

Data quality issues are most sever when information is scattered across isolated and heterogeneous data stores. To turn information into insight and to leverage its significant value, data quality needs to be addressed by applying data cleansing consistently, using consistent cleansing rules throughout the enterprise, not only in the database layer but also in the application and process layers as mentioned by Sautter and Mathews [7].

The Pattern analysis is aimed to understand which formats are valid for a particular field by examining the distribution of character types found within the column. If we analyze the patterns over several thousand records we can start to uncover the data quality rules for this field as the patterns that occur with the most frequency are typically the valid rule and the ones which occur least frequently are often in error.

The main advantages of data cleansing patterns are the increment of data quality, the reusability of the rules in order to update data that are not under the specified pattern and therefore inaccurate, besides of lower costs of data maintenance [7].

C. Record Matching

Deterministic matching systems use a combination of algorithms and business rules to determine when two or more records match. Algorithms catch simple common errors such as typos, phonetic variations and transpositions. Either the records match the requirements of the business rule or they do not match. However, deterministic systems lack scalability. When the amount of records increases, deterministic matching requires expensive customization and

business rule revision, impacting performance and cost as a consequence.

IV. FRAMEWORK OF DATA CLEANSING WITHIN MOBILE DEVICES

In the case of mobile applications, the information most frequently used is concerned with personal information such as contact details, audio data, e-mails, notes, tasks to do, etc. There is not massive information like in data warehouse environments but sufficiently relevant for making business.

This section details a Framework proposed and implemented for Data Cleansing within mobile devices. The main objective of this Framework is to cover the main issues of data quality under a very pragmatic focus. Fig. 1 shows the most relevant steps.

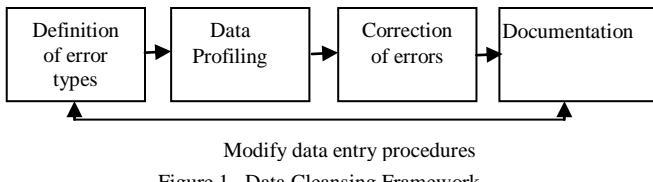


Figure 1. Data Cleansing Framework.

A. Define and determine error types

Regarding the quality properties within mobile devices, we present the most frequent problems of accuracy, completeness, usability, usefulness. The definition of such quality properties has been pointed in section III. Therefore, the corresponding error types are mentioned as follows:

a) Accuracy: Within the context of data stored in mobile devices, inaccurate data would be a misspelled contact name, which in fact may incur in a business failed, or a misspelled name song.

b) Amount of data: In the case of mobile devices, the possibility to detect and correct wrong data by hand becomes impractical at the range of hundreds.

c) Format appropriateness: In the case of IDv3 tag would be better to homogenize format by detecting and forcing pattern specified by users.

d) The Format precision there are values correctly represented, values not represented (missing) and values that do not correspond with real world. The data cleansing process should support several character sets in order to be suitable for different languages.

e) Representation consistency: In the case of contact details there must exist data integrity constraints that force the corresponding format or data pattern in order to avoid data errors. For instance, in the case of songs, the author name should be an aggregation of first name, last name, etc.

f) The Uniqueness property: Frequently after and upgrade or migration of contacts we find our data duplicated because of different formats between one mobile device and the new one. A similar situation is in the case of downloaded mp3 songs, because there are different versions of IDv3tags.

g) Usability: Nowadays is usable a software that allows to edit details of a business contact or the details of a mp3 song, but when the manipulation of data is in terms of hundreds, the software becomes unusable at all.

h) Usefulness: Wrong data would not be useful for the task at a hand.

B. Data profiling

This process required analysis and clustering phases for parsing and searching in order to identify data quality patterns and error instances in the source files and then isolation of these data elements in target files.

- Analysis

Given a data set of a number of mp3 files the prototype gets the ID3 tag from the end of each mp3 file and returns whether there was an ID3 tag on the file, throws unsupportedEncodingException if system cannot handle ASCII or throws IOException if an error occurs reading the file.

- Clustering

The prototype examines file by file and extracts all metadata of each file such as title, artist, year, album, comment and filename then save these values into a text file.

The process of extracting metadata of mp3 files in IDv3 tags format was achieved by adjusting the JID3 library proposed in [7]. During this phase the prototype groups the mp3 files by artist name in a number of .txt files and explores which are the fields that have null values to mark them as candidates for data consolidation.

- Data Quality Patterns

Regarding the analysis and detection of data patterns, there has been a considerable work for a number of data repositories such as Excel, Access, Oracle and SQL Server [8],[19] but not for IDv3 tags. Therefore, we have implemented a data quality pattern analyzer in order to obtain the greatest percentage of occurrences for each column of the mp3 files metadata.

The development of a data quality pattern was through the mapping of standard ASCII character set to letters that indicate their character format. For example, the lowercase letters have been mapped to 'L' meaning lowercase standard alphabetical character. Uppercase letters have been mapped to 'U'. Numbers were mapped to 'N' meaning number.

The simplest form for performing the mappings is to convert the ASCII number of a character to the corresponding mapping character as many of the characters are not visible on a standard editor.

The conversion function converts each character to the new mapping. After obtaining the format pattern for each item, the data quality pattern counts different patterns occurrences in order to identify which pattern is the most frequent utilized and define it as the pattern to be forced by the integrity constraint.

In the case of the IDv3 tags, some of the data quality patterns are shown in Table 1. For instance, the song title will be forced to follow the U*WU*WU* pattern because the 45% of the song names are capitalized.

The patterns for each field of the IDv3 tag are stored in a metadata file, in order to avoid the analysis of data patterns each time more mp3 files are loaded to the mobile device.

TABLE 1 DATA PATTERNS DETECTED FOR IDV3 TAG

Title_Sample	Pattern	%
amor de voceador	L*WL*WL*	11
Amor de voceador	UL*WL*WL*	13
AMOR DE VOCEADOR	U*WU*WU*	45
Amor_de_Voceador	UL*SL*SUL*	9
AMOR_DE_VOCEADOR	U*SU*SU*	10
Amor-de-Voceador	UL*DL*DUL*	12

Once the patterns are established for each field of the tag the next step is the correction of errors by the data matching, data consolidation, and data standardization.

C. Correction of errors

The process of correction of data errors required sophisticated data algorithms and secondary data sources.

The prototype explores all metadata within each txt file and measures strings distance to detect redundancy and ensure consistency or to detect duplicated records and ensure uniqueness.

- Data Standardization

In order to ensure data consistency, the prototype applies conversion routines to transform data into its preferred (and consistent) format.

For instance, the audio file can be renamed accordingly to the user pattern detected. For instance, Artist – Title (comment), Title (comment), Artist – Title (comment), Title (comment), etc.

- Data Matching:

Data matching is the process of searching and matching records within and across the parsed, corrected and standardized data based on predefined business rules to eliminate duplications [10].

The present work applies the JaroWinklerDistance class [6] in order to identify duplicated songs when there are a relatively strong proximity between their corresponding song titles and album names.

For each artist text file a comparison of each record is done by measuring the distance between titles and albums of all songs if there is similitude of 80% between them or more, we assume that these records are the same entity of the real world.

An example of a simple comparison is shown in the following code:

```
for(int z=x+1;z<canciones.length;z++){
if(!canciones[x].getTitulo().equals("null")&&!canciones[z].getTitulo().equals("null")){
if(jaroWinkler.proximity(canciones[x].getTitulo(),
canciones[z].getTitulo())>0.8)
```

- Data consolidation:

This is a process of consolidating or merging matched records into one representation. However, in order to achieve completeness and remove redundancy, a comparison between the matching records is carried out in order to preserve existing data values by replacing the corresponding null values for the same field. Finally, the mp3 files are renamed with the complete title of the song following the pattern detected from the user, and with complete and accurate information.

D. Documentation

The patterns identified from user regarding the IDv3tag information are documented and stored in order to keep standardization and to restrict and correct information modifying future data entry procedures.

E. Modify data entry procedures to reduce future errors

Every time a new song or contact is captured within the mobile device the patterns shall be maintain.

The documentation and the modification of data entry procedures are part of our work in progress.

V. TEST AND EXPERIMENTATION

Regarding the appropriateness of the proposed framework, we have identified representative scenarios according to the data quality properties assessed on specific data sources to test whether the data profiling and the correction of errors are within the expected ranges.

Table 2 shows some tests considering 10, 20, 45 and 77 mp3 files and the execution time the prototype takes according with the number of mp3 files cleaned.

The elapsed time is mainly taken during the process of reading the mp3 files and the process of writing the information in text files.

The record matching and cleaning are not taking a long time to process. However, more detailed and exhausted testing of the framework needs to be carried out.

TABLE 2 MP3 FILES AND EXECUTION TIME

Amount of files	ms
10	4984.0
20	13750.0
45	21188.0
77	42078.0

Fig. 2 shows an example of original mp3 files contained within a folder named "My Music". There were a number of data quality errors: (a) duplicated songs (Aclaraciones), (b) different format representation for Artist name (FDO DELGADILLO, FERNADO DELGADILLO, Fernando Delgadillo, etc.), (c) inaccuracies for song titles (mía, día, pirámide), and (d) incompleteness derived from missing information in album name and artist name fields.

Nombre	Intérprete	Título del álbum
02 EL ABORDAJE	FERNANDO DELG...	
03 Aclaraciones	Fernando Delgadillo	DE VUELOS Y DE SOL
09 LLOVIZNA	FERNANDO DELG...	DE VUELOS Y DE SOL
14 - ME AND MY SHADOW	Robbie Williams	SWING WHEN YOU'RE ...
A la pirámide del sol	Fdo. Delgadillo	
A TU VUELTA	Fdo. Delgadillo	Entre Paireos y Derivas
Aclaraciones	Fernando Delgadillo	DE VUELOS Y DE SOL
ADVERTISING SPACE	ROBBIE WILLIAMS	Intensive Care
ANGELS	ROBBIE WILLIAMS	Greatest Hits
AUNQUE NO TE VUELVA A VER	ALEX UBAGO	REALIDAD O SUEÑO
Carta a Francia		FEBRERO 13 VOLUMEN 2
COME UNDONE	Robbie Williams	Escapology
CUENTO	FERNANDO DELG...	Feb 13 VOL 2
El adios	FERNANDO DELG...	Primera Estrella
ENTRE PAIROS Y DERIVAS	FERNANDO DELG...	
Eres mía	FERNANDO DELG...	Vol.1
EVOLUCIONES	FERNANDO DELG...	FEBRERO 13 VOLUMEN 1
FEEL	Robbie Williams	Greatest Hits
Fernando Delgadillo - COSAS ...		
Fernando Delgadillo - OLVIDAR	fernando delgadillo	15 Super Hits
Hoy hace un buen día		FEBRERO 13 VOLUMEN 2
HOY TEN MIEDO DE MI	FERNANDO DELG...	

Figure 2. Example of original mp3 data.

Fig. 3 shows the mp3 data after the data cleansing process. The patterns detected for the song name, artist and title of album have been forced to represent data consistently. Duplicated mp3 files have been merged for completeness and uniqueness.

Nombre	Intérprete	Título del álbum
A LA PIRÁMIDE DEL SOL	FERNANDO D...	
A TU VUELTA	FERNANDO D...	ENTRE PAIR...
ACLARACIONES	FERNANDO D...	DE VUELOS Y ...
ADVERTISING SPACE	ROBBIE WILL...	INTENSIVE C...
ANGELS	ROBBIE WILL...	GREATEST HITS
AUNQUE NO TE PUEDA VER	ALEX UBAGO	REALIDAD O ...
CARTA A FRANCIA	FERNANDO D...	FEBRERO 13 ...
COME UNDONE	ROBBIE WILL...	ESCAPOLOGY
COSAS Y PALABRAS	FERNANDO D...	
CUENTO	FERNANDO D...	FEB 13 VOL 2
EL ABORDAJE	FERNANDO D...	
EL ADIOS	FERNANDO D...	PRIMERA EST...
ENTRE PAIROS Y DERIVAS	FERNANDO D...	
ERES MÍA	FERNANDO D...	VOL.1
EVOLUCIONES	FERNANDO D...	FEBRERO 13 ...
FEEL	ROBBIE WILL...	GREATEST HITS
HEAVEN	ROBBIE WILL...	
HOY HACE UN BUEN DÍA	FERNANDO D...	FEBRERO 13 ...
HOY TEN MIEDO DE MI	FERNANDO D...	

Figure 3. Mp3 files after data cleansing process.

VI. CONCLUSION AND FUTURE WORK

We have proposed and mostly implemented a Data Cleansing Framework based on previous work in order to

deal with poor data quality properties, such as accuracy, amount of data, format precision, representation consistency, duplicated values, and usefulness of data and usability of software.

We present a prototype with the implementation of the method proposed by Maletic and Marcus [3] and the removing technique explained by Loshin [2] applied to mobile devices and enhanced to cope with poor data quality.

Data quality patterns and data matching algorithms have been implemented and enhanced for the detection and correction of data errors within the process of data cleansing in mobile devices.

Data cleansing has being widely used on data warehousing, but not on data stored in mobile devices such as mobile phones, electronic organizers o audio players. The prototype developed within the present research is able to clean mp3 files only with id3tag v1 and v2.

Finally, we are designing a set of test for a full range of mobile devices in order to prove portability and to the correct function of the prototype.

As part of our future work, the cleansing algorithms implemented should support a wider range of formats. Supporting different character sets, a full implementation of data cleansing for all id3tag and vCard versions.

In the case of completeness we require to establish which data sources would be suitable for connection trough internet for further information that could be merged for missing information.

ACKNOWLEDGMENT

This work was supported by a grant from Dirección General de Asuntos del Personal Académico, UNAM.

REFERENCES

- [1] P. Angeles M. and F. Garcia Ugalde "A data quality practical approach". The International Journal on Advances in Software, IARIA.ISSN: 1942-2628, 2009, Vol. 2 No. 2&3, pp. 259-274.
- [2] P. Angeles M. and F. Garcia Ugalde, "Assessing Quality of Derived Non Atomic Data by considering conflict resolution Function", First International Conference on Advances in Databases, Knowledge, and Data Applications,2009, 978-0-7695-3550-0/09 IEEE, pp. 81-86.
- [3] S. Chaudhuri and U. Dayal. "An overview of data warehousing and OLAP technology". SIGMOD Rec. 26, 1 (March 1997), 65-74. DOI=10.1145/248603.248616.
- [4] E. Rham and H. H. Do, "Data Cleaning: Problems and Current Approaches" in the Bulletin of the Technical Committee on Data Engineering, December 2000 Vol. 23 No. 4 IEEE Computer Society, pp.3-13.
- [5] A. E. Monge "Matching Algorithms Within a Duplicate Detection System", in the Bulletin of the Technical Committee on Data Engineering, December 2000 Vol. 23 No. 4 IEEE Computer Society, pp.14-20.
- [6] D. Loshin "Enterprise knowledge management", Chapter 3 and 4, pp. 17-37 and pp. 48-52.
- [7] J. Maletic and A. Marcus. "Data cleansing: beyond integrity analysis". Division of Computer Science. The Department of Mathematical Sciences. The University of Memphis. Campus Box 526429.

- [8] M. Hernandez and S.J. Stolfo. "Real-World data is dirty: data cleansing and The Merge/Purge problem". Department of Computer Science Columbia University New York, NY 10027.
- [9] P. Anokhin and A. Motro, "Fusionplex: Resolution of Data Inconsistencies in the Integration of Heterogeneous Information Sources", Technical Report ISE-TR-03-06, Information and Software Engineering Dept., George Mason University, Fairfax, Virginia, 2003
- [10] Class JaroWinklerDistance: <http://alias-i.com/lingpipe/docs/api/com/aliasi/spell/JaroWinklerDistance.html> (retrieved; November 17, 2010.)
- [11] G. Sauter and B. Mathews, "International Bussiness Machine (IBM), Information service Patterns", Part 3 Data cleansing patern, 2007. <http://www.ibm.com/developerworks/webservices/library/ws-soa-infoserv3/> (retrieved; November 17, 2010.)
- [12] Mp3Tagger java program Stephen Ostermiller, Copyright © 2007 Free Software Foundation, Inc., <http://fsf.org/> (retrieved; November 17, 2010.)
- [13] L. Apicella, "Innovative Research Strategies: The Use of Handheld Computers to Collect Data the Population Council" <http://www.popcouncil.org/horizons/ORToolkit/toolkit/pda.html> (retrieved; November 17, 2010.)
- [14] S. Schumacher, "Probabilistic Versus Deterministic Data Matching: Making an Accurate Decision", Information Management Special Reports, January 2007
- [15] C. Redman, "Data Quality for the Information Age", Boston, MA., London : Artech House, 1996, pp. 551-582
- [16] R.K. Bock and W. Krischer,"The Data Analysis BriefBook" Springer 1998.
- [17] V. Barnett and T. Lewis, "Outliers in Statistical Data", John Wiley & Sons, New York, 1984.
- [18] R.B. Buchheit, "Vacuum: Automated Procedures for Assessing and Cleansing Civil Infrastructure Data", PhD Thesis, May 2002
- [19] A.F. Karr, A.P. Sanil, and D.L. Banks," Data Quality: A Statistical Perspective", Technical Report 151, March 2005, National Institute of Statistical Sciences.
- [20] Y. Lee and D. Strong, "Knowing-Why about Data Processes and Data Quality", Journal of Management Information Systems, Vol. 20, No. 3, pp. 13 – 39. 2004.
- [21] Y. Wand and R. Wang, "Anchoring Data Quality Dimensions in Ontological Foundations", Communications of the ACM, Vol. 39, No. 11, pp. 86-95, 1996.
- [22] R. Y Wang and D.M. Strong, "Beyond accuracy: What data quality means to Data Consumers", Journal of Management of Information Systems, Vol. 12, No. 4 1996, pp. 5 -33.
- [23] Data Quality Pro Forum, <http://www.dataqualitypro.com/> (retrieved; November 17, 2010.)

Remote Comparison of Database Tables

Fabien Coelho CRI, Maths & Systems, MINES ParisTech,
35 rue Saint Honoré, 77305 Fontainebleau cedex, France.

fabien.coelho@mines-paristech.fr

Abstract—Database systems hold mission critical data in all organizations. These data are often replicated for being processed by different applications as well as for disaster recovery. In order to help handle these replications, remote sets of data must be compared to detect unwanted changes due to hardware, system, software, application, communication or human errors. We present an algorithm based on operations and functions already available in relational database systems to reconcile remote tables by identifying inserted, updated or deleted tuples with a small amount of communication. A tree of checksums, which covers the table contents, is computed on each side and merged level by level to identify the differing keys. A prototype implementation is available as a free software. Experiments show our approach to be effective even for tables available on a local network. This algorithm provides a communication efficient and general solution for comparing remote database tables.

Keywords—remote set reconciliation; data replication.

I. INTRODUCTION

Relational database systems must hold reliably mission critical information in all organizations. These data are often stored in multiple instances through synchronous or asynchronous replication tools or with bulk data transfers dedicated to various transactional and decisional applications. These data replications can help enhance load sharing, handle system failures, application, software or hardware migrations, as well as applicative transfers from one site to another.

As trust does not preclude control, it is desirable to compare the data between remote systems and identify inserted, deleted or updated tuples, to detect errors or to resynchronize data. This is known as the set reconciliation problem. Few differences are expected between both data sets. Key issues include big data volumes, site remoteness and low bandwidth. Transferring the whole data for comparison is not a realistic option under these assumptions.

This paper presents a generic algorithm to compare relational database tables, which may reside on remote and heterogeneous DBMS such as open source PostgreSQL and MySQL or proprietary Oracle and DB2. We assume that the compared tables are composed of records identified by a key, say the primary key of the relation. We do not make other assumptions on the data sets, such as the availability of attributes that may be used for grouping data, or restrict the number, type, size or value of attributes involved in the comparison either as the key or as the other columns. The algorithm finds the key of inserted, updated or deleted tuples with respect to a parametric subset of rows (part of the records are investigated) and columns (the comparison is restricted to some attributes). It relies on simple SQL constructs and functions available on all systems. Summaries are extracted

on each server and transferred to the client system where the reconciliation is performed. A block parameter allows to optimize the latency/bandwidth trade-off.

The paper is organized as follows. Section II details the related work. Then, Section III presents the comparison algorithm and the SQL queries performed to build the necessary checksum tree and compute the differences. The algorithm is then analyzed in Section IV and discussed in Section V. Section VI describes our implementation. Experiments are reported in Section VII. Finally, Section VIII concludes our presentation.

II. RELATED WORK

Suel and Nemon [1] analyze many comparison algorithms for delta compression and remote synchronization. A first class of problem addresses locally available data sets, and targets identifying deltas. Chvatal *et al.* [2] described string to string combinatorial research problems in 1972. Solutions are found for many of these problems, but these approaches do not apply to our problem as they deal with locally available data sequences. A second class deal with data stored on remote locations, and aims at identifying missing or differing parts without actually transferring the data. The remote set comparison problem has been addressed with various techniques. A key issue is whether the data are naturally ordered, such as a string or a file composed of pages, or considered as a set of distinct unrelated elements, *i.e.*, unordered data, such as the files in a file system or the tuples of a relation.

Metzner [3] introduced a binary hash tree reconciliation to compare file contents. Our approach can be seen as an extension to handle tuple keys that identify records as opposed to the intrinsic page numbering that come with a file, and we use a parametric group size to select better trade-off opportunities depending on multiple optimization factors such as bandwidth, number of requests or disk I/Os. The practical rsync algorithm [4][5] is well known to system administrators. It is asymmetrical in nature. Blocks of data already available on one side are identified and complementary missing data are sent to the other side. Block shifts are identified at the byte level thanks to a sliding checksum computation. With respect to our problem, such approach could result in easy identification of inserts, but very poor network performances for updates and deletes.

Coding-theory based solutions [6][7][8] reduce the number of communication rounds and the amount of transfers for comparing remote data sets. The key idea is that as the data are already available with very few differences, only the error correcting part of a virtual transmission is sent

and allows to reconstruct the differences. Such techniques need significant mathematical computations on both sides that are not readily available with the standard SQL functions of relational database systems.

Descending recursive searches based on hashes over subsets on both sides were proposed [9], which is similar to [10] discussed later. Bloom filters are also used to statistically reduce the amount of data to communicate [11] in a synchronization, however this structure results in more false positive especially when the number of differences in the reconciliation is small. The same paper also briefly alludes to a Merkle tree [12], similar to Metzner and to our approach, for computing set differences.

In all of the above remote set comparison techniques, the differing elements are extracted either as inserts or deletes, but updates are not detected as such: as there is no concept of key to identify elements, updates cost two searches in the complexity computations to be identified as an insert and a delete. A few papers address our problem from the same viewpoint of (1) having a key to identify elements, (2) distinguishing updates from inserts and deletes and (3) being able to perform the algorithm through simple SQL queries.

Maxia [13] presents several asymmetric algorithms to compare remote tables using checksums. One level of summary table is used, leading to an overall communication complexity in $\mathcal{O}(k(b+n/b))$ where k, n, b are the number of differences, the table size and a block size. The algorithm for inserts and updates does not detect deletes and does not operate properly in some limit cases, whereas the algorithm for deletes does not work if other operations were performed. Ideally, a solution should detect all kind of differences with a low communication complexity. This will be our focus.

Schwartz [14][15] presents a top-down checksum approach to detect and update out-of-sync replication nodes. The algorithm uses the DBA knowledge in order to group tuple checksums according to attribute values, and thus may take advantage of existing indexes, but providing such a beneficial information is described as tricky by the author.

Vandiver [10] also presents a top-down *N-round-X-rows-hash* reconciliation algorithm for remote databases. The checksum approach is similar to ours, with the difference that the hash tree is not materialized, and by relying on an existing integer primary key. The first round computes hashes for records grouped in buckets based on the primary key value. Each differing bucket hash is then investigated by subdividing it, up to the N-th round where individual rows are considered. The algorithm can take advantage of existing indexes in the primary key, but it also requires some tuning. It performs best when defects are highly correlated.

Finally, software products are also available to identify differing rows, such as DBDiff [16] or DB Comparer [17]. Although these tools compare table contents, the actual algorithm used and its bandwidth requirements are unclear. Snapshots suggest a graphical user interface, which displays differing data to help the user perform a manual reconciliation. Such packages also focus on table structure comparisons, so as to derive SQL ALTER commands necessary to shift from one relational schema to another.

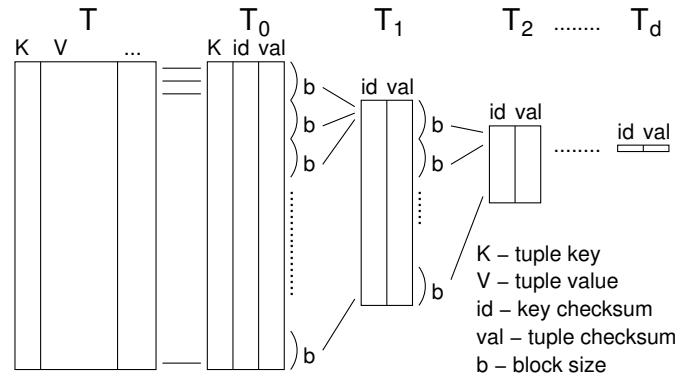


Fig. 1. Initial table T , checksum table T_0 and tree of summary tables $T_{i,i \geq 1}$

```
CREATE TEMP TABLE T0
AS SELECT
    K AS key,
    h(K) AS id,
    h(K,V) AS val
FROM T WHERE W;
```

Fig. 2. Checksum table T_0 with hash h

```
CREATE TEMP TABLE Ti
AS SELECT
    id&mi AS id,
    XOR(val) AS val
FROM T0
GROUP BY id&mi;
```

Fig. 3. Summary tables $T_{i,i \geq 1}$

III. REMOTE COMPARISON ALGORITHM

Let us now present the hierarchical algorithm for comparing two remote database tables named T with K the primary key, V the attributes to be compared, and W a condition to select a subset of the rows to be analyzed.

The algorithm is fully symmetrical. It computes on both sides a hierarchical tree of summaries shown in Figure 1. They are then scanned from the root downwards to identify the differing tuple keys by investigating the differences concurrently. The reconciliation is achieved by merging the summaries at each level. It does not decide what to do with the differences, but simply locates the offending keys and reports them. A natural continuation may be to transfer the offending data so as to re-synchronize the tables.

First, a checksum table T_0 storing both tuple keys and signatures is built as shown in Figure 2, using hash function h . Second, aggregations in Figure 3 compute the summary tree for all the tuples through a set of reduction masks $m_{i,i \geq 1}$. The last table, T_d , only holds one summary checksum for the whole table. Third, remote selects in Figure 4 on the summary tables allows to reconcile the tuples and thus to identify inserts, updates and deletes by a merge algorithm that deals with key and value checksums in Figure 6. It first compares the one row of table T_d . If they are different, it walks through the tree to check for the source of the differences up to the checksum

```
// get checksums at level i
list getId(c, i, what)
withkey = (i==0)? "", key": ""
return sql2list(c, "
    SELECT id, val $withkey
    FROM Ti
    WHERE id&mi+1 IN ($what)
    ORDER BY id $withkey")
```

Fig. 4. Summary extraction query

```
// show a block
// of matching keys
showKeys(c, msg, l)
for v,i in l
    for key in sql2list(c, "
        SELECT key FROM T0
        WHERE id&mi = $v")
        print "$msg $k"
```

Fig. 5. Key extraction query

```

// reconcile on connections c1 c2
merge(c1, c2)
list curr = (0), next, ldel, lins;
level=d;
while (level>= 0 and curr)
  // get checksums at level
  list lid1 = GetIds(c1, level, curr),
  list lid2 = GetIds(c2, level, curr);
  // merge both sorted lists
  while (i1 or i2 or lid1 or lid2)
    i1,v1,k1 = shift(lid1) if no i1;
    i2,v2,k2 = shift(lid2) if no i2;
    if (i1 and i2 and i1==i2)
      // matching key checksum
      if (v1!=v2)
        // differing value checksum
        if (level==0) print "UPDATE $k1"
        else append i1 to next
      elseif (no i2 or i1<i2)
        // single checksum in lid1
        if (level==0) print "INSERT $k1"
        else append (i1,level) to lins
        undef i1
      elseif (no i1 or i1>i2)
        // single checksum in lid2
        if (level==0) print "DELETE $k2"
        else append (i2,level) to ldel
        undef i2
    level--;
    curr = next
// whole block differences
showKeys(c1, "INSERT",lins)
showKeys(c2, "DELETE",ldel)

```

Fig. 6. Reconciliation merge algorithm

table where the actual tuple keys are available. If a whole checksum block is empty on one side, the corresponding keys are extracted with a special query outlined in Figure 5.

The set of mask used in the aggregation is built as follows: Let n be the table size (number of rows in T), h a checksum function, possibly cryptographic. f the folding factor logarithm (that is the block size for folding is $b = 2^f$). Let $\ell = \lceil \ln(n)/\ln(2) \rceil$ be the closest power of 2 above the table size, $d = \lceil \ln(n)/\ln(b) \rceil$ be the tree depth, then $m_i = 2^\ell/b^i - 1 = 2^{\ell-if} - 1$ (when $1 \leq i \leq d$) is the grouping mask for level i , with $m_d = 0$.

It is important that m_1 should reduce the size of the first table by the folding factor, otherwise some folding factor values result in bad performances because the first folded table T_1 is not folded and takes as much time to compute as checksum table T_0 . This is pointed out in Figure 7-3 of [10]. The last folding may be less efficient, but it is negligible as the data volume involved is very small at the root of the tree.

IV. ANALYSIS

Let us analyze the above algorithm with respect to the disk I/Os, computations and communications involved. We will use the following additional notations: l the tuple length (size of attributes), k the number of differences to be found, c the number of bits of checksum function h .

The amount of computation and I/O performed by the database depends on the optimizations implemented by the

query processor and on the data size. The main cost is incurred in building the initial summary table, all $\mathcal{O}(nl)$ data of the initial data must be read, and $\mathcal{O}(nc)$ data must be written for the checksum table. Moreover, heavy computations are involved at this stage as two checksums are computed on the data read, which may also involve various data conversions depending on the actual data types. These checksums can be maintained as additional columns with triggers so that they would be available directly. Then, the first aggregation can be performed with a merge sort in $\mathcal{O}(n \ln(n))$, or an hash technique done on the fly in $\mathcal{O}(n)$, and the subsequent ones are reduced by power of b leading to an overall $b/(b-1)$ factor. The final requests for the merge phase just require to scan some data, which may be helped by indexes to be created possibly in $n \ln(n)$ operations. Thus, the overall computation cost on each side is $\mathcal{O}(nl + n \ln(n) \cdot b/(b-1))$.

The number of requests of the client-server protocol depends whether there are differences: An initial request gets the table size necessary to compute the tree depth and the relevant masks; Then, one batch of queries can build the checksum, summary tables, and return the initial root summary checksum. If they match, the tables are equals, otherwise the reconciliation must dig into the tree up to the leaves. Thus there are up to $\mathcal{O}(\ln(n)/\ln(b))$ requests.

The amount of data communicated at each stage depends on the selected block size and the number of differences to be found. For small k , $\mathcal{O}(\lceil \ln(n)/\ln(b) \rceil \ln(n))kcb$ data are communicated: each differing id of size $\ln(n)$ is investigated on the depth of the tree, and each found block to be merged contains cb bits. As k grows, a steady state is reached when all blocks at level 0 are scanned as they all contain at least one difference: the communication is then $\mathcal{O}(cnb/(b-1))$. This steady state comes around $k = n/2^f$. Such a saturation effect is encountered in the third set of experiments presented in Section VII.

There is a latency/bandwidth trade-off implied by the choice of the block folding factor: the higher b the higher the amount of data to be transferred, but the lower the number of requests as the depth is reduced.

V. DISCUSSION

Our algorithm reduces the amount of data to be communicated through two key points: First, the attributes are compacted together with a checksum, which will usually be smaller than the data it summarizes. A checksum collision at this level would result in differences not to be found. Second, rows are aggregated together by building a summary tree, and only some of the computed records will be needed for the comparison. There again, a collision of the computed values at any level of the summary tree could result in differences not being found.

A key hypothesis is that few differences are expected: otherwise the search process scans most of the table through many requests, although an ordered scan of the initial table would allow to identify the missing or differing items in a single pass. If the hypothesis is not met, the implementation may allow the user to stop the computation when the number of differences encountered is above a given threshold.

One checksum computation is performed on the key and another on the key and value attributes. The first hash aims at randomizing the key distribution so that aggregations group tuples evenly, and so that the computations do not depend on the key type and composition. It is also needed to differentiate updates from inserts and deletes. A simple integer evenly distributed key may be used instead if available. The second hash of the key and value part identifies the items. The key in the second checksum is necessary and is not redundant with the previous hash: if not included, a simple exchange of values between two tuples would not be detected if they are aggregated in the same group.

As usual with checksum functions, their size (c bits) and quality should be good enough to avoid collisions. A collision on the key hash in the checksum table makes two tuples identified as one, thus a difference detected on one would be reported about the other as a false positive, but this is easily filtered out by checking the tuple key also available in the checksum table. A collision of the value hash of two tuples in the checksum table does not have an impact on the search because tuples are also identified by their key hash. A collision of *both* the value hash for the same key hash on the checksum table would result in a difference not to be reported, thus leading to a false negative. A collision of the value hash in the summary tables for the same level and group would result in differences within these tables to be ignored. Cryptographic hash functions with $c \geq 128$ make such collisions improbable.

The tree of aggregations computes a common checksum by combining tuple chunks. This operation should treat individual checksum bits equally. Exclusive-or (XOR) is the usual operator of choice if available. If not, the SUM aggregation can be considered, provided it applies to the checksum result type. Note that the central limit theorem is not an issue for the SUM aggregation: the more deterministic bit values are confined to upper bits of the sum, possibly removed by modulo arithmetic, while the lower bits only are significant in the hash combinations, and those stay as random as the inputs. The group criterion should also be compatible with the checksum result and allow to define tuple chunks. In order to compute directly the group of a tuple at any level, its computation must only depend on the level and not on the groups computed at the preceding levels. This property is achieved by a binary mask or a modulo operations on the power of an integer.

The algorithm is fully symmetrical. Inserts and deletes are only parted on the convention that the first table serves as the reference in the comparison, but the structure of the algorithm is the same on both sides. The algorithm handles missing intermediate keys with two special lists, `lins` and `ldel`. This case arises if a whole chunk of tuples is removed or added.

As noted by Maxia [13], it is possible to maintain the checksum directly in the initial table or in another by mean of trigger procedures, so that they would not need to be computed over and over. It could also be integrated in the database as a new kind of hash index dedicated to remote table comparison.

It must be noted that the checksum and each summary tables are built and then queried once: thus computing an index to help access some data is not beneficial as its building cost would not be amortized.

VI. IMPLEMENTATIONS

We know of three implementations of our algorithm. We developed a proof of concept free software prototype [18] targeting both PostgreSQL and MySQL, including actually synchronizing tables between both systems. Vandiver [10] developed a version in Java in order to compare his algorithm with ours in his PhD thesis, but the corresponding code is not available. Finally, Nacos [19] built a C-coded trigger-based tool using one big summary table, while the reconciliation algorithm is also in Java.

When considering an implementation of our algorithm, several key functionalities are needed: the ability to replace NULL values, a checksum function, a grouping criteria and a relevant aggregate function.

First, as NULL values propagate through SQL functions, they must be dealt with in order to keep meaningful checksums: SQL COALESCE function can be used to substitute NULL values. Second, we use a shortened version of the MD5 cryptographic hash function, which is available on both systems as a checksum. However, in order to have results comparable between PostgreSQL and MySQL, a lot of hopus-pocus was necessary for casting, converting and truncating the results reliably. We added cast functions to PostgreSQL and developed special conversion functions for MySQL. As the comparison may be CPU bound, especially on a local network, we also provide a fast although not cryptographically secure checksum function as extensions to both database systems. Third, a criterion must be chosen to group the tuples in the tree building phase, compatible with the data type holding the key checksum. Our implementation store results as 8-byte integers, and we restricts block sizes to power of 2, so that we can use simple mask operations. Finally, a checksum combining aggregate function must be provided. Our implementation uses either SUM or XOR.

The table comparison can be restricted to perform partial checks: The comparison to be performed is specified through URL-formed command line arguments with reasonable defaults, and an option can select a subset of tuples with a WHERE clause.

Another implementation issue is whether to use threads to parallelize the queries on both sides, especially the initial checksum table building which represent the largest computation cost. This can influence significantly the performance up to a factor of two on a fast network. However, on a slow network, most of the time is spent in communications with a bandwidth shared by the two parallel connections, and the impact on performance is small or null. Our perl script

```

SELECT COALESCE(T1.key, T2.key) AS key,
CASE WHEN T1.key IS NULL THEN 'DELETE'
      WHEN T2.key IS NULL THEN 'INSERT'
      ELSE 'UPDATE'
END AS operation
FROM T1 FULL JOIN T2 ON ((T1.key)=(T2.key))
WHERE T1.key IS NULL          -- DELETE
      OR T2.key IS NULL        -- INSERT
      OR (T1.val) <> (T2.val) -- UPDATE

```

Fig. 7. Local comparison of tables T1 and T2 (with NOT NULL attributes)

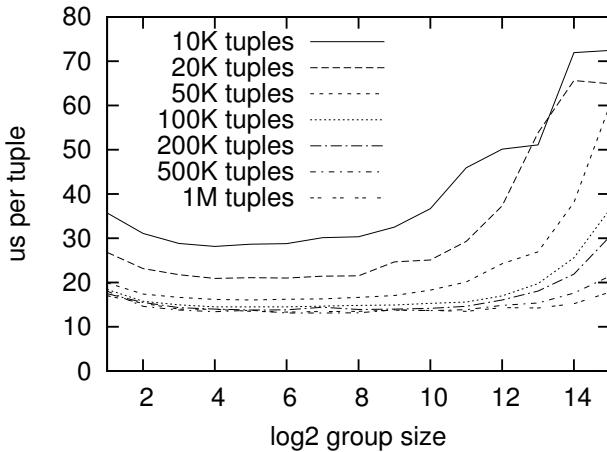


Fig. 8. Comparison time per tuple, 1 Gb/s, 3 diffs

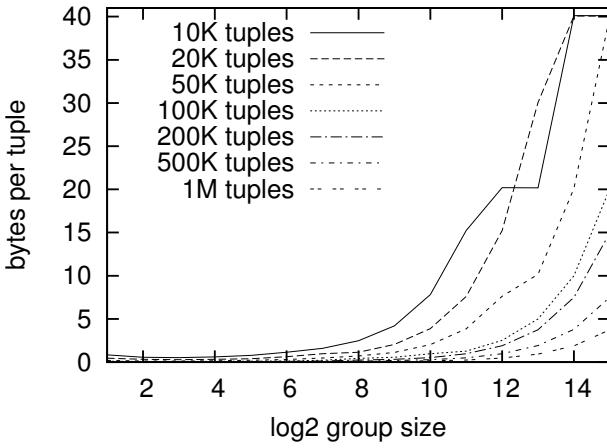


Fig. 9. Volume transferred per tuple, 1 Gb/s, 3 diffs

implementation includes a threading option, but it does not work with the PostgreSQL because of a bug in the standard driver implementation.

VII. EXPERIMENTS

We report in the following about 5500 runs of our comparison algorithm performed with our implementation on PostgreSQL databases using the fast checksum, on randomly generated tables from 10 K top 1 M rows of 450-byte long records, compared with varying block sizes, and in a bandwidth bound environment. Three Linux desktop computers in an isolated network were used, two of them holding the databases, the third performing the reconciliation. The network bandwidth was controlled on every connection with the `tc` traffic control command. The base case for a remote comparison algorithm would be to transfer all tuples on the other side, and then perform the comparison locally with an external join as suggested in Figure 7.

The following measures must be taken with caution: they are sensitive to many parameters such as hardware including hard disk, cpu and memory performance, system disk cache management, database configuration and optimization, algorithm implementation details such as threading, network

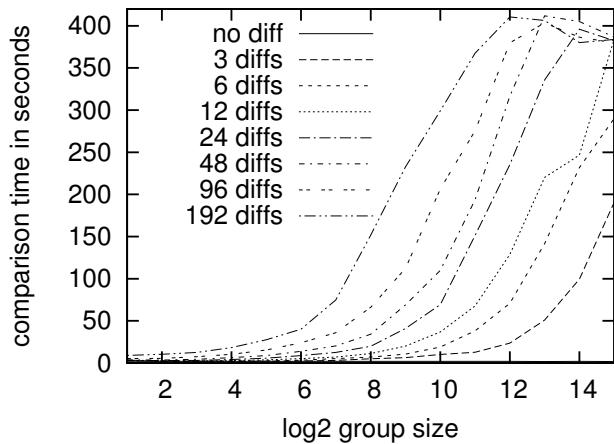


Fig. 10. Comparison time vs block size, 100 K tuples, 100 Kb/s bandwidth

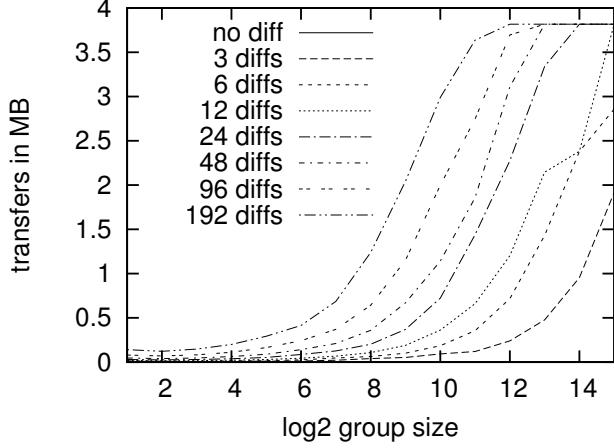


Fig. 11. Volume transferred vs block size, 100 K tuples

latency, bandwidth, mtu and congestion status, as well as various protocol overheads. To encompass all these, we used overall elapsed times: it covers both computation and network, the preeminence of which varies depending on the actual test conditions.

Figures 8 and 9 show normalized data collected from a 1 Gb/s local area network. The horizontal axis is the log₂ group size used for building the summary tree. The vertical axis in the normalized time to perform the reconciliation in μ s per tuple for the first figure, and the number of bytes per tuple (without overheads) transferred for the second. Different table sizes are investigated to recover 3 differences. The two small table sizes incur high latency and summary computation overheads. For other larger sizes, the comparison requires a somehow constant 14 μ s/tuple. As there are few differences, the main costs are in computing the checksum and summary tables and in network latency, especially for small group sizes. The drilling down is mostly negligible but for large group sizes where big chunks of checksums are fetched. The base case of transferring data alone is about 4.5 μ s/tuple in this setting, thanks to the high bandwidth.

In Figures 10 and 11, a 100 Kb/s network link is used to reconcile 100 K tuple tables with different block parameters.

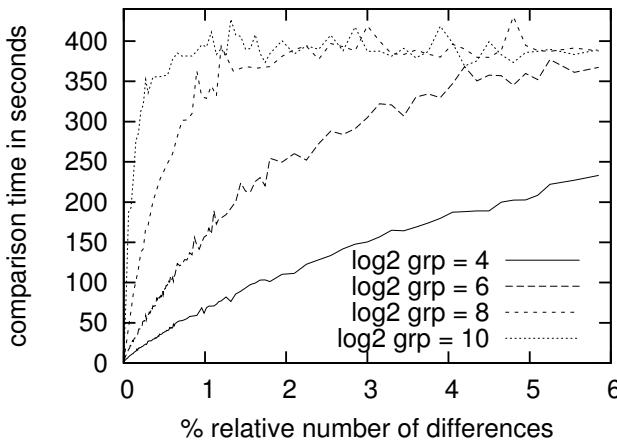


Fig. 12. Comparison time vs diffs, 100 K tuples, 100 Kb/s bandwidth

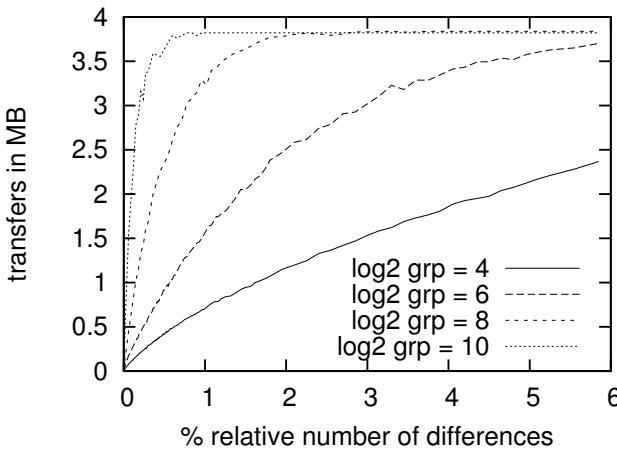


Fig. 13. Volume transferred vs diffs, 100 K tuples

The transfer of all data would require about 4000 seconds, so the 2 seconds comparison time achieved is a 2000-fold speedup. The number of differing tuples is made to vary from 0 to 192, and the comparison time is displayed in seconds. Block sizes from $8 = 2^3$ to $128 = 2^7$ perform best. Larger block values are loaded by the requirement in network bandwidth and as the number of differences increases. There is a saturation when all checksums are fetched from T_0 .

A 100 Kb/s bandwidth network is also used in Figures 12 and 13. The number of differences is scaled up for various block sizes and 100 K tuples, and shown as a percentage relative to the table size. The linear dependency of the algorithm in a network bound context shows up for small number of differences, and for higher figures the saturation effect is encountered when all checksum blocks are fetched. The turn around for saturation is expected when all checksums are fetched, that is at about size $n/2^f$, i.e., 6%, 1.5%, 0.4% and 0.1% for the four block sizes presented. The base case in this setting is about 1 hour to transfer the table, while 7 minutes suffice to identify the differences based on checksums.

Vandiver [10] also performed experiments with his implementation of our algorithm, and compared them to his own: The main costs comes from computing the checksum table.

The drilling through the data structures requires few communications. When considering highly correlated faults, our generic approach can be beaten because the key randomization in groups becomes a liability.

VIII. CONCLUSION

We have presented an algorithm dedicated to remote relational database table comparisons with a parametric block size. Keys of inserted, deleted or updated tuples are identified quickly. This algorithm can be implemented on top of reasonable instances of SQL: most of the checksum work is performed through database requests, and the client tool performs a reconciliation merge of partial checksums fetched level by level. All experiments of our remote comparison algorithm show better performances than the brute force download solution, but for the Gb/s local network. This shows that our implementation provides a generic, elegant and portable solution to remote comparison of database tables.

Acknowledgment – Thanks to Laurent Yeh for pointing out relevant related work, and to Michael Nacos and Benjamin M. Vandiver for local or remote discussions.

REFERENCES

- [1] T. Suel and N. Memon, *Lossless Compression Handbook*. Academic Press, 2002, ch. Algorithms for Delta Compression and Remote File Synchronization.
- [2] V. Chvatal, D. A. Klarnet, and D. E. Knuth, “Selected combinatorial research problems.” Stanford University, Tech. Rep., 1972.
- [3] J. J. Metzner, “A parity structure for large remotely located replicated data files,” *IEEE Trans. Computers*, vol. 32, no. 8, pp. 727–730, 1983.
- [4] A. Tridgell and P. MacKerras, “The rsync algorithm,” Australian National University, TR-CS 96-05, Jun. 1996.
- [5] A. Tridgell, “Efficient algorithms for sorting and synchronization,” Ph.D. dissertation, Australian National University, 1999.
- [6] K. A. S. Abdel-Ghaffar and A. E. Abbadi, “An optimal strategy for comparing file copies,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 5, no. 1, pp. 87–93, 1994.
- [7] M. Karlovsky, L. Levitin, and A. Trachtenberg, “Data verification and reconciliation with generalized error-control codes,” *IEEE Transactions on Information Theory*, vol. 49, no. 7, pp. 1788–1793, Jul. 2003.
- [8] Y. Minsky, A. Trachtenberg, and R. Zippel, “Set reconciliation with nearly optimal communication complexity,” *IEEE Transactions on Information Theory*, vol. 49, no. 9, pp. 2213–2218, Sep. 2003.
- [9] A. Trachtenberg and Y. Minsky, “Efficient Reconciliation of Unordered Sets,” Cornell University, Tech. Rep. 1778, Nov. 1999.
- [10] B. M. Vandiver, “Detecting and Tolerating Byzantine Faults in Database Systems,” Ph.D. dissertation, MIT, Cambridge, MA, USA, Jun. 2008, MIT-CSAIL-TR-2008-040.
- [11] J. W. Byers, J. Considine, M. Mitzenmacher, and S. Rost, “Informed content delivery across adaptive overlay networks,” *IEEE/ACM Transactions on Networking*, vol. 12, no. 5, pp. 767–780, Oct. 2004.
- [12] R. C. Merkle, “Secrecy, authentication and public key systems / a certified digital signature,” Ph.D. dissertation, Stanford University, 1979.
- [13] G. Maxia, “Taming the distributed database problem: A case study using MySQL,” *Sys Admin*, vol. 13, no. 8, pp. 29–40, Aug. 2004.
- [14] B. Schwartz, “MySQL toolkit: mk-table-sync,” <http://www.maatkit.org>, Mar. 2007, checked 2010-11-11.
- [15] ———, “An algorithm to find and resolve data differences between mysql tables,” Blog on <http://www.xaprb.com>, Mar. 2007, checked 2010-11-11.
- [16] DKG Advanced Solutions, “DBDiff for windows,” <http://www.dkgas.com/>, 2004, checked 2010-11-11.
- [17] EMS, “Db comparer,” <http://www.sqlmanager.net/>, 2006, checked 2010-11-11.
- [18] F. Coelho, “PgComparator,” Software available on <http://pgfoundry.org/projects/pg-comparator>, Aug. 2004, version 1.7.0 on 2010-11-12.
- [19] M. Nacos, “Pg51g,” <http://pgdba.net/pg51g/>, Sep. 2009, checked 2010-11-11.

An Approach for Distributed Streams Mining Using Combination of Naïve Bayes and Decision Trees

Meng Zhang

School of Computer Science
Beijing University of Technology
Beijing, China
therealzhangmeng@gmail.com

Guojun Mao

College of Information
Central University of Finance and Economics
Beijing, China
maximmao@hotmail.com

Abstract—Nowadays we have various kinds of data generated at high speed in distributed environment. In many cases, it is difficult or unallowed to gather all the distributed data into a central place for processing. So we have to perform part of the work at the location where data is generated. In this paper, we present an approach for mining distributed data streams by using combination of naïve Bayes and decision tree classifiers. The method takes advantage of both distributed and stream mining characteristics. Each local site uses ensemble classifiers of decision tree to learn a concept of incoming stream and transmits local pattern to a central site. The central site combines the collected patterns to build a global pattern using an attribute-weighted strategy in order to relax the attribute independent constraint of naïve Bayes model. The experiment shows that by using this approach we can get comparable understanding of the global data while reducing transmission load and computation complexity.

Keywords-data stream mining; distributed streams

I. INTRODUCTION

Nowadays, one of the main research directions in data mining field is data stream mining. Because it is common that real world applications produce streaming data, such as transactions in banking, stock market and e-commerce. Among these learning tasks distributed environment is also a common context due to the widely utilization of the Internet. Data stream mining has some characteristics different from traditional data mining. The data is generated at high speed, arrived continuously and potentially infinite. So it challenges the storage, computation and communication capabilities of the computer systems.

By examining previous related work in data stream mining, we found that the train of thought in many contributions is to get inspiration from traditional mining and extend the idea to fit the streaming context. Distributed mining context is a loosely coupled environment. Each element in the system can do some independent work without affect other ones. We think this trait is proper to combine separate classification methods and bring advantages of both methods into play.

Naïve Bayes [7] and decision tree [10] are two of the most widely used induction algorithms for classification tasks. Decision tree is a common learning technique which gives easy interpretation of data and performs well in many learning fields. Naïve Bayes classifier has a simple

assumption that attributes are independent of each other. Although this assumption is often violated in real learning tasks, naïve Bayes classifier can get comparable prediction accuracy with other methods in many domains.

Since naïve Bayes classifier was introduced, there are many researches on improving this method. Langley and Sage [8] introduce an approach of using only a subset of the attributes to make predictions and they show it can improve accuracy in domains with redundant attributes. Elkan [3] applies Boosting to naïve Bayes classifier which builds several classifiers instead of only one. Many people have worked on combining decision tree with naïve Bayes to get better prediction results. Kohavi [6] builds decision tree containing naïve Bayes classifiers in the leaves which is called NBTree. He points out that this hybrid frequently outperforms both constituents in larger databases. Ratanamahatana and Gunopulos [11] propose a method selecting attributes that appear in the nodes of decision trees and using only these attributes to do naïve Bayes learning. Though their approach gets better results than C4.5 decision tree and naïve Bayes classifier on many datasets, it needs several scan of the training set.

Most of the improved methods are done under the condition of traditional mining on static datasets. Due to the characteristics of streaming data, it is necessary to use different strategies to do the mining task. Domingos and Hulten [2] describe a very fast decision tree (VFDT) learning algorithm based on Hoeffding trees for streaming data. They build a decision tree at the beginning and refine it incrementally when new data is coming. Street and Kim [12] propose an ensemble method called SEA to mining data stream. This method uses many relatively small decision trees instead of maintaining one large decision tree, so it can refine the model more efficiently than incrementally maintained single tree.

In distributed mining field, Parthasarathy et al. [9] introduce a distributed stream processing architecture. In this architecture, each distributed computing node learns a local model and a central site uses these local models to generate a final model. They point out that this method provides both parallelism and scalability. Sun et al. [13] present a hierarchical algorithm for summarizing several local patterns into a global pattern which requires only a single pass over the data.

In this paper, we extend the approach of combining decision tree and naïve Bayes classifier to distributed data

streams mining environment. We build an ensemble of C4.5 decision trees at each local site and compute statistical summary of each data chunk. Then we select some attributes that are the split attributes at top levels of the C4.5 decision trees in the ensemble as key attributes. The local pattern is made up of these key attributes and the statistical summary. Each local site transmits its local pattern to the central site and the central site builds a new naïve Bayes classifier based on the local patterns to maintain the global pattern. By transmitting local patterns rather than raw data from local sites to central site, we can achieve a lower traffic cost in distributed environment. Besides, building naïve Bayes from statistical summary is more efficient than learning it from raw data.

The rest of this paper is organized as follows. In Section 2, we describe our method for combining decision tree and naïve Bayes in distributed streams mining task. Section 3 shows the experiment results and analysis. Section 4 discusses the conclusions of our method and points out some future work.

II. LOCAL SITE MINING PROCESS

Each local site continuously receives data, and the received data is buffered for a special time interval to form a data chunk. When a chunk is available for processing, we will do three things on it as follows:

Firstly, a statistical information structure for this trunk, called *statistical summary*, is constructed. For a discrete attribute, we count the number of instances per attribute value per class. For a numeric attribute, we calculate sum and quadratic sum per attribute value per class. When computing the distribution of attribute values in a trunk, the number of instances per class in this trunk can also be gotten. Here we use a matrix to store the *statistical summary*.

$$\begin{bmatrix} & c_1 & c_2 & \cdots & c_n \\ a_1 & \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1n} \end{bmatrix} \\ a_2 & \begin{bmatrix} s_{21} \\ \vdots \\ \ddots \\ \vdots \end{bmatrix} \\ \vdots & & & & \\ a_m & \begin{bmatrix} s_{m1} & \cdots & s_{mn} \end{bmatrix} \end{bmatrix}$$

Figure 1. Matrix to store *statistical summary*.

In the matrix, each column represents a class value and each row represents an attribute. Each cell of the matrix is called a *summary record*. For a discrete attribute, e.g. attribute a_1 has d distinct values, the *summary record* s_{11} is an array of length d with each cell stores number of instances which have that specific value on a_1 and take the class value c_1 . For a numeric attribute, e.g. attribute a_2 , the *summary record* s_{21} is a structure made up of the sum and quadratic sum of attribute a_2 's value of all the instances that take the class value c_1 . Using matrix to store the *statistical summary* is convenient and proper because we can compute the global model by simply doing matrix addition of the *statistical summary* from all the local sites.

Secondly, a C4.5 decision tree is generated for the current data chunk, and use the current data chunk as a test set to validate the previous ensemble. During the process, we test each classifier in the ensemble on the newly arrived data chunk and also the newly constructed decision tree. Then we use the average classification accuracy of these classifiers and a threshold bound to decide whether the current tree is appropriate to add to the ensemble. Only if the accuracy of the current tree is higher than the average accuracy minus threshold bound, it can be added to the ensemble. If the ensemble is not full, we directly insert the new C4.5 decision tree into the ensemble. Otherwise, according to the testing result, the decision tree in the ensemble that gets the worst result on the current data chunk will be replaced by the new tree that is just constructed. Here, we take a simple replace tactic to refine the ensemble classifiers. This is because that the newest tree can represent the new changes of the data stream better.

Thirdly, the key attributes in the ensemble are selected. Given an integer k , the attributes locate on top k levels of all trees in the ensemble are considered as key attributes, and called as *Top-k-level-attributes*. Research like [11] has proved that the attributes that are located closer to the root are more important than others in a decision tree, and so we just need to pick those attributes appear at the top k levels of the decision trees. Take ensemble classifying into consideration, an *impact factor* need to be set for each attribute of each tree, which is inversely proportional to its location in the tree. That is, the root of a tree has the maximal *impact factor* value; the closer a node is located to the root, the larger its attribute's *impact factor* is. After key attributes are selected, if an attribute of them appears on several trees in the ensemble, we calculate the average value of the *impact factor* in these trees as the value of *impact factor* for this selected attribute. As far as an attribute *impact factor* is concerned, it is directly related to the levels that this attribute locates on all local decision trees. We can assign 2^k to the *impact factor* of the root attribute, 2^{k-1} to the *impact factor* of the attribute locates at second level of the tree, and so on. By doing this, the difference of *impact factor* between two levels is relatively obvious. For those attributes are not key attributes, their *impact factor* are assigned to 1.

In our design, the local pattern of a local site is made up of the *statistical summary* and the *Top-k-level-attributes*. Figure 2 gives description of mining a local pattern in the local site.

```

D: the current data chunk in the data stream
E: an ensemble of C4.5 decision tree classifiers
T: a C4.5 decision tree classifier in the ensemble
Tc: the current learned C4.5 decision tree
Tw: the worst decision tree in the ensemble

repeat when there are more data chunks in the stream
    read one data chunk D
    compute statistical summary of D and store it in a matrix structure
    construct a decision tree Tc based on D
    use D to evaluate all decision trees in the ensemble E and also the Tc
    if Tc is appropriate to add to ensemble
        if E is not full
            insert Tc into E
        else
            find the worst tree Tw in E
            replace Tw with Tc
        end
    end
    for each tree T in E
        select out top-k-level-attributes
    end
    calculate the impact factor of these attributes
    transmit local pattern to central site
end

```

Figure 2. Local site process algorithm.

III. GLOBAL MODEL CONSTRUCTION

When a local site finishes processing a data chunk, it will transmit its own local pattern to the central site. As soon as the central site receives the local patterns from all the local sites for a data chunk, it will start generate the global pattern on this time. In our design, these local patterns have the same structure. Generating the global pattern is divided into two steps. Firstly, we need to build a naïve Bayes model from the *statistical summary* which is part of the local pattern. Secondly, we use attribute weights to adapt naïve Bayes model to improve classification performance.

We know that naïve Bayes classifier comes from the Bayes' rule of probability theory. It also has a core assumption that attribute is independent of each other. So based on these two conditions, we can describe naïve Bayes model as:

$$P(C_i | a_1 a_2 \dots a_n) = \frac{P(a_1 | C_i) P(a_2 | C_i) \dots P(a_n | C_i) P(C_i)}{P(a_1 a_2 \dots a_n)} \quad (1)$$

In the formula, C_i is for the value of class i and a_j is for attribute j . Since for a given data instance, the denominator is the same regardless of its class value, we can just consider only the numerator part of the formula. So our task is just to determine the probability of each attribute for a class value from local *statistical summaries*.

There are two types of attributes, discrete and numeric. For a discrete attribute, the probability $P(a_k | C_i)$ can be estimated by the proportion of instances that have the Attribute a_k to the number of instances in Class C_i . The probability $P(C_i)$ can be estimated by the proportion of instances in Class C_i to the size of investigated data. For numeric attribute, it is common to assume a normal distribution model. Thus, the probability $P(a_k | C_i)$ and $P(C_i)$ can be determined from the distribution function of Attribute a_k and Class C_i , respectively. All the information we need to calculate the probability is contained in the local *statistical summaries*.

In order to relax the attribute independent assumption, we introduce attribute weights, and adjust the numerator part of the Bayes formula as follows:

$$P(a_1 | C_i)^{w(a_1)} P(a_2 | C_i)^{w(a_2)} \dots P(a_n | C_i)^{w(a_n)} P(C_i) \quad (2)$$

Where $w(a_i)$ is the weight of Attribute a_i . Note that the range of the probability is between 0 and 1. So the less the weight, the more impact the related attribute has. And we use a simple mathematical function to transform the impact factor of attribute to its weight:

$$f(x) = \alpha^x, \quad 0 < \alpha < 1 \quad (3)$$

By doing this, the key attributes take greater value of impact factor, so they get smaller value of weight, which in turn have greater probability result and will blow their importance up in decision process. From another point of view, attribute dependencies can be partly eliminated by using attribute weights. Figure 3 describes the process of global model construction at the central site.

```

repeat receive local patterns from local sites
    sum the matrices of the local patterns up to
    get statistical summaries of the whole data at
    current chunk
    construct naïve Bayes model based on all
    the statistical summaries
    transform the impact factor of all the
    attributes to generate their weights in the
    model
    use this weighted Bayes model with to do
    classification on global data
end

```

Figure 3. Global model construction algorithm.

IV. EXPERIMENTAL RESULTS

We did the experiment in a simulated distributed environment and the sites are implemented as multiple threads in the program. Though we chose three local sites and one central site for keeping simplicity, our design can be applied to more complicated example easily. We used WEKA [5], the famous open source software for data mining,

and two datasets from UCI repository [4] in the experiment. We chose the datasets based on some criteria, e.g. containing many attributes with both discrete and numeric, having large number of instances. The datasets that we selected both contain tens of thousands of instances.

The Adult dataset [6] has information for task of determining whether a person makes over 50K a year. It includes 14 attributes with 6 of numeric type and 8 of discrete type.

The Forest CoverType dataset [1] has information for predicting forest cover type from cartographic variables. It includes 54 attributes and most of them are discrete. It has 7 distinct classes.

In the experiment, we first randomly repeated the data and shuffled it to generate sufficient amount of data. Then we split the dataset into three parts, each for a local site. We continuously read the dataset at a different rate to simulate the data stream. We tested the classification accuracy at the central site. As a comparison, we centralized all the data at the central site to do a single stream mining using ensemble C4.5 decision tree classifiers only.

We did the experiment with some parameters. Each data chunk is composed of 500 instances and the minimal interval between the arrivals of two instances is set to 10 milliseconds. Take into account the number of attributes in each dataset, we set the parameter k of *top-k-level-attributes* to 2 with Adult dataset and 3 with CoverType dataset.

Figure 4 and 5 shows the accuracy result of our experiment with Adult dataset and CoverType dataset respectively. In these plots, we can see the accuracy of distributed mining method is lower than the accuracy of centralized mining method most of the time. Sometimes the

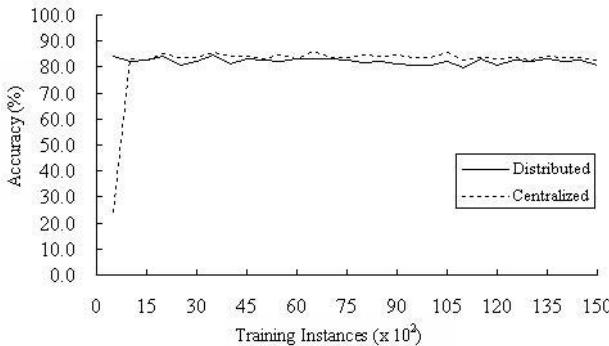


Figure 4. Result of Adult dataset with 500 instances per chunk.

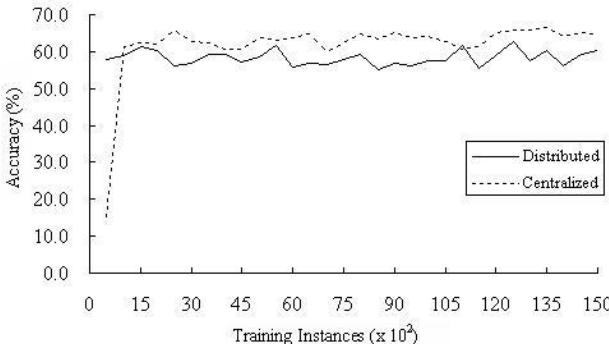


Figure 5. Result of CoverType dataset with 500 instances per chunk.

distributed result can outperform the centralized method. On all accounts the difference between them is not much. So we can say the performance of our approach is comparable with the centralized method.

We also record the number of *top-k-level-attributes* that selected from each data chunk. Figure 6 indicates that on average only 7 out of 14 attributes for Adult dataset were selected as key attributes. Figure 7 shows that on average 25 out of 54 attributes for CoverType dataset were selected as key attributes. Concentrating on these key attributes improves our global classification model.

V. CONCLUSION

We have described a method of using C4.5 decision tree and naïve Bayes classifiers in distributed stream mining. This is to extend the combination of decision tree and Bayes learning. It takes advantage of the characteristic of both decision tree and naïve Bayes model. Decision tree is easily interpreted and has hierarchical structure which can distinguish some important attributes from others. By using ensemble of decision trees, we can eliminate the tree pruning work which will slow down the processing efficiency. And it will not degrade the performance because the ensemble method can combine several weak classifiers to get good results. Naïve Bayes classifier can be trained efficiently given the *statistical summary*. It is particularly suited when the dimensionality of the dataset is high like in our experiments. The experimental result shows that it is an applicable approach for its simplicity and efficiency.

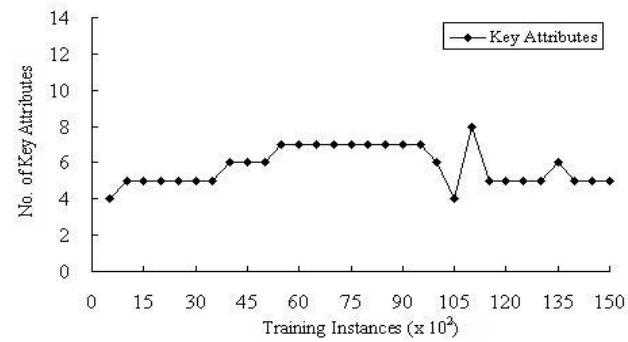


Figure 6. Key attributes selected from Adult dataset with 500 instances per chunk.

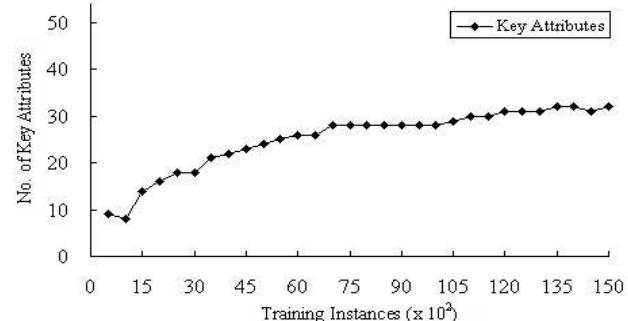


Figure 7. Key attributes selected from CoverType dataset with 500 instances per chunk.

The information technology develops and changes fast. This work indicates that researching on some of the classical mining tasks can give us inspirations and we can use them for reference when the target or context changes.

In future research, we will focus on making our approach more general to other datasets. That is, datasets with less amount of data, less number of attributes, etc. Besides, handling the drift of concept is another issue to think about.

ACKNOWLEDGMENT

This research has been supported by Beijing Municipal Key Laboratory of Multimedia and Intelligent Software Technology and the National Science Foundation of China under Grants No.60873145.

REFERENCES

- [1] J. Blackard and D. Dean, "Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables," *Computers and Electronics in Agriculture*, vol. 24(3), Elsevier Press, Dec. 1999, pp. 131-151, doi:10.1016/S0168-1699(99)00046-0.
- [2] P. Domingos and G. Hulten, "Mining high-speed data streams," Proc. The 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 00), ACM Press, Aug. 2000, pp. 71-80, doi:10.1145/347090.347107.
- [3] C. Elkan, "Boosting and naïve bayesian learning," Technical report, Department of Computer Science and Engineering, University of California, San Diego, 1997.
- [4] A. Frank and A. Asuncion, "UCI Machine Learning Repository [<http://archive.ics.uci.edu/ml>]," Irvine, CA: University of California, School of Information and Computer Science, 2010.
- [5] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. Witten, "The WEKA data mining software: an update," *SIGKDD Explorations*, vol. 11(1), 2009.
- [6] R. Kohavi, "Scaling up the accuracy of naïve-bayes classifiers: a decision-tree hybrid," Proc. The 2nd International Conference on Knowledge Discovery and Data Mining (KDD 96), AAAI Press, Aug. 1996, pp. 202-207.
- [7] P. Langley, W. Iba, and K. Thompson, "An analysis of bayesian classifiers," Proc. The 10th National Conference on Artificial Intelligence, AAAI Press, Jul. 1992, pp. 223-228.
- [8] P. Langley and S. Sage, "Induction of selective bayesian classifiers," Proc. The 10th Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Press, Jul. 1994, pp. 399-406.
- [9] S. Parthasarathy, A. Ghoting, and M. Otey, "A survey of distributed mining of data streams," *Data streams: models and algorithms*, C. C. Aggarwal, Springer Press, 2007.
- [10] J. Quinlan, "C4.5: programs for machine learning," Morgan Kaufmann Press, 1993.
- [11] C. Ratanamahatana and D. Gunopulos, "Scaling up the naïve bayesian classifier: using decision tree for feature selection," Proc. Workshop on Data Cleaning and Preprocessing (DCAP 02), at IEEE International Conference on Data Mining (ICDM 02), Maebashi, Japan, 2002.
- [12] W. Street and Y. Kim, "A streaming ensemble algorithm (SEA) for large-scale classification," Proc. The 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 01), ACM Press, Aug. 2001, pp. 377-382, doi:10.1145/502512.502568.
- [13] J. Sun, S. Papadimitriou, and C. Faloutsos, "Distributed pattern discovery in multiple streams," Proc. The 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 06), Springer Verlag Press, Apr. 2006, pp. 713-718.
- [14] L. Wang, X. Li, C. Cao, and S. Yuan, "Combining decision tree and naïve bayes for classification," *Knowledge-Based Systems*, vol. 19(7), Elsevier Press, Nov. 2006, pp. 511-515, doi:10.1016/j.knosys.2005.10.013.

Systems Biology Warehousing: Challenges and Strategies toward Effective Data Integration

Thomas Triplet

Centre for Structural and Functional Genomics
Department of Computer Science
Concordia University
1455 De Maisonneuve Blvd. West, Montreal, QC Canada
Email: thomastriplet@gmail.com

Gregory Butler

Centre for Structural and Functional Genomics
Department of Computer Science
Concordia University
1455 De Maisonneuve Blvd. West, Montreal, QC Canada
Email: gregb@encs.concordia.ca

Abstract—The rapid development of genomics, proteomics, metabolomics and structural genomics techniques have provided an unprecedented amount of data, enabling system-wide biological research. Although information integration has been well investigated in database theory research, biological data present numerous challenges from the lack of standard formats to data inconsistencies resulting from experimental data variations. Satisfying and practical solutions are still lacking and current molecular biology databases serve primarily as simple data repositories with limited query capabilities. They also provide little to no integration with other databases. However, the success of systems biology is contingent on the ability to integrate and utilize a wide variety of types of data. It also relies on computational techniques to automatically predict and assign functional annotations of proteins as effective integration of biological data should enable scientists to perform comparative analyses, modelling and inference of protein functions. Therefore, there is a need for a paradigm shift toward systems biology databases with flexible query systems that focus on answering a diversity of questions from biologists without the need to reconfigure the underlying database architectures.

Keywords-genomics; proteomics; systems biology; data warehousing; data integration

I. INTRODUCTION

Life sciences techniques made significant improvements over the past decades, resulting in huge amounts of data collected over the years by the scientific community. In order to facilitate the organization and the subsequent analyses of this valuable data, databases have been developed very early. Since then, the number of databases has dramatically increased. The 2010 Molecular Biology Database Collection [1] includes well over a thousand databases, each describing millions of biological records.

This unprecedented wealth of information originating from genomic studies represents a tremendous potential in all areas of biological science. The emerging information integrated with existing knowledge bases could lead to an explosive understanding of complex molecular interactions, networks and pathways. Successful data integration is one of the keys to successful bioinformatics research [2]: scientists need an integrated view of these heterogeneous data sources

with advanced data-mining, analysis and visualisation tools. The continuing data growth will lead to an increasing need for large-scale data management as biological discovery depends, to a large extent, on the presence of clean, up-to-date and well-organised datasets.

Unfortunately, the rapidly growing number of different molecular biology databases, created at various places worldwide, serve primarily as data warehouses with simple query interfaces designed for specific tasks. The databases are not readily amenable to complex system-based research that requires the integration of a large number of these disparate databases. There is a need for a paradigm shift toward systems biology databases with flexible query systems that focus on answering a diversity of questions from biologists without the need to reconfigure the underlying database architectures.

In this paper, we present an overview of the problem of the integration of multiple biological databases from the perspective of large-scale analysis of biological systems. Section II overviews some technical aspects of data warehousing. Section III explains the specificities of biological data and some of the challenges they raise when being integrated, from data heterogeneity (Section III-A) to experimental variability (Section III-C). Section IV then describes data warehousing requirements for effective systems biology and reviews key features and limitations of several major data warehousing frameworks.

II. DATA INTEGRATION METHODS

Information integration has been well investigated in database theory. Currently, three main approaches are generally considered when integrating data: the *Extract, Transform and Load* (ETL), the *Local-As-View* (LAV) and *semantic integration* methods.

A. Extract-Transform-Load method

An ETL-based warehouse is constructed by Extracting, Transforming and Loading the data to integrate into a single unified schema. The transformation step allows the data to

be pre-processed before they are integrated in the warehouse, which may be useful to address some of the problems mentioned in Section III, in particular those related to data inaccuracies and inconsistencies. However, ETL methods generally lack flexibility because the warehouse schema is tightly coupled to the data sources. As a result, integrating new databases requires considerable effort as the entire warehouse and subsequent queries need to be redefined. The warehouse schema may also have to be redesigned if one of the data sources schema changes after an update.

B. Local-As-View method

Local-As-View (LAV) methods [3], [4] are designed to address the flexibility issues of ETL methods. They are based on functions — or wrappers — that provide an abstraction layer and a simplified view of the integrated data sources. They traditionally rely on dynamic logical views, which are featured by most DBMSs. However, logical views are usually constructed using natural joins to correlate database keys and can therefore provide erroneous or misleading answers (see Section III-C for details). Dynamically generating views that contains millions of records can also be computationally expensive and/or inefficient.

C. Semantic integration

Unlike the ETL and LAV methods, semantic integration methods [5], [6] do not address issues related to the underlying architecture of the DBMS of the warehouse, but focus on the *semantic* integration of related entities or concepts from heterogeneous data sources through the use of ontologies, that is, formal descriptions of the concepts and entities for a domain of interest and the relationships that hold among them. Semantic integration is therefore useful to handle heterogeneous data that lack standardization (see Section III-B) as if often the case in biology. It is however computationally expensive and often requires large data centres to be effective [7].

III. BIOLOGICAL DATA SPECIFICITIES AND CHALLENGES

Biological data present a number of specificities and raise many challenges when being integrated, in particular in the context of large-scale analyses of biological systems as a whole. As a result, satisfying and practical solutions derived from the methods described above have proven to be elusive for these complex data sources.

A. A wealth of heterogeneous data

The 2010 Molecular Biology Database Collection [1] contains 1,230 databases, requiring a database of databases to integrate data and keep track of all the knowledge available to biologists today [8]. Among these databases is the extent of our knowledge related to genomics [9], proteomics [10], metabolomics [11] and structural genomics [12].

As an illustration of the vast amounts of accessible data, consider: (i) the RefSeq [13] database, containing 16 million sequences (152 billion base pairs) from over 10,000 species, (ii) the Joint Genome Institute (JGI) sequencing projects [14], comprising 200 billion base pairs, (iii) the Gene Ontology [15], containing 30,914 terms that describe the biological function of nearly 500,000 gene products (iv) the eggNOG (evolutionary genealogy of genes: Non-supervised Orthologous Groups) database [16], describing 224,847 orthologous groups covering 2.5 million proteins (v) the Kyoto Encyclopedia of Genes and Genomes (KEGG) [17], containing 116,962 metabolic pathways comprising 16,291 compounds.

While these five distinct molecular biology databases represent a small fraction of those available, they still contain a wealth of data beneficial to system biology studies. But, conducting searches or identifying correlations across just these five databases ranges from extremely limited to non-existent. Furthermore, the web-based interfaces are generally designed with the expectation of presenting the user with only a few results from a query. The display and manual analysis of queries that generate hundreds or thousands of results is not practical.

B. Lack of unique standards

Integrating data from multiple databases provides an additional challenge because of the different data formats and structures [18]. Those data are very disparate and often stored in database management systems (DBMS) or as simple plain text files in multiple data repositories, which provide very limited compatibility or interoperability between systems.

Describing a chemical structure clearly illustrates this critical problem since there are more than 80 different formats currently in use. There are also numerous ways to name a chemical compound, which includes common names, abbreviations and systematic nomenclatures [19] defined by the International Union of Pure and Applied Chemistry (IUPAC). IUPAC systematic nomenclatures are an evolving process and tend to be cumbersome for relatively complex molecules and difficult to generate even with software that automate the process. Errors are common and there is a relatively high failure rate in actually generating a name. An alternative approach uses a linear character string to represent a 2D structure of the compound. SMILES (Simplified Molecular Input Line Entry System) is the most popular approach of generating a simple text-based representation of a chemical structure [20]. However, numerous distinct and correct SMILES strings can be generated from a single chemical structure. InChI (International Chemical Identifier) — “a new standard for molecular informatics” — is a recent variation on SMILES strings [21]. While InChI generates a unique character string for a given molecule, it is relatively new and not as widely used as SMILES strings.

Another example is DBGET/LinkDB [22], which aims at providing a unified database query mechanism. Yet, only a handful databases are capable of handling DBGET/LinkDB requests. Furthermore, using DBGET, it is currently not possible to query two databases and automatically integrate the results: data from the integrated databases remains largely independent and difficult to combine.

C. Data inconsistencies and inaccuracies

1) *Experimental variations:* Because of their experimental nature, biological data are also routinely sparse and/or inaccurate. For example, consider the fungus *Aspergillus niger* available from two major repositories for fungal genomics: the *JGI Genomes* [23] database developed by the U.S. Department of Energy and *EnsemblFungi* [24], a fungal database maintained by the European Bioinformatics Institute. GBrowse [25] is the primary means to visualize genomes on JGI whereas EnsemblFungi is powered by BioMart [26].

On JGI Genome, consider the gene fgenesh1_pg.C_scaffold_6000331 located on Chromosome I:612-2,561, which codes for a protein involved in fungal specific transcription. The sequence of this gene was matched with sequences from EnsemblFungi using BLASTN [27]. The best hit on EnsemblFungi was a perfect match ($E = 0$, 100% sequence identify). However, on EnsemblFungi, the gene is located on Chromosome III:3,638,575-3,640,524. Similarly, gene e_gw1.12.166.1 at the locus Chr.III:2,768-4,422 on JGI was identified ($E = 1.5 \times 10^{-88}$) at Chr.II:2,593,707-2,594,464 on EnsemblFungi. In addition, both repositories use non-standardized mapping systems: coordinates are relative to scaffolds in JGI whereas they are relative to chromosomes in EnsemblFungi.

2) *Data entry errors:* Compounding the above inconsistencies are entry errors in the original data, such as spelling mistakes and the inadvertent addition or deletion of characters or spaces [28]. A number of experimental errors associated with the data are also to be expected [29], where estimates of annotation errors for gene products range from 8% to 49%, depending on the method [30]. Similarly, spelling and typographical errors have been measured to occur at rates of 1.5 to 2.5% and 1 to 3.2%, respectively [31]. Thus, data errors present an inherent challenge in the development of molecular biology databases [32].

3) *Approximate string matching and similarity functions:* 3rd Millennium® showed that the number of incorrect entries grows geometrically with the number of joins and reaches nearly 50% by the fourth join, assuming a conservative error rate of 15% per join [2]. Solely relying on database indexes to correlate database keys is generally dangerous and blind data integration can provide erroneous or misleading answers. Similarity functions unique to

molecular biology data are therefore required and numerous similarity algorithms have been developed but these are generally implemented as independent stand-alone programs accessible through web-servers. Nearly 1,200 web links to resources and software accessible to the scientific community have been documented [33]. This provides a variety of valuable tools to improve the quality and flexibility of biological database searches. For instance, BLAST [27] and FASTA [34] are well-known and highly utilized sequence homology approaches that search sequence databases using substitution matrices and string matching heuristics. Alternatively, PSI-BLAST [35] uses a profile-sequence comparison method, and HMMER applies hidden Markov models [36]. Similarly, programs such as Dali [37] and Structural [38] align the 3D structures of proteins present in the Protein DataBank. The Espresso [39] program combines both sequence and structure alignments to identify similar proteins.

Errors in the data are more likely to be accommodated by the robustness of these similarity searches [40] and the most reliable approach is to incorporate similarity algorithms into the database structure to *simulate* indexes that are normally based on binary search trees or hashes in most DBMS. Similarity measures may also improve semantic integration [41] when combined with ontologies.

Moreover, spelling mistakes may be detected using approximate string matching algorithms [31]. In particular, Damerau [42] and Levenshtein [43] estimated that 80% of human spelling mistakes could be automatically corrected using at most one character insertion, deletion, substitution or transposition.

4) *Data provenance:* To help address data quality issues, tracking the provenance of the data is critical [44]. The provenance typically consists of metadata associated with the data and is helpful for scientists to evaluate their quality and reliability. It also allows them to examine the lineage of a piece of information, which shows all the steps involved in sourcing, moving and processing the data. Toward that end, all datasets and their transformations must be recorded. Goble [45] suggested that data provenance could be useful to address ownership and copyright issues as well as to record experimental protocols followed to generate the data, effectively ensuring the reproducibility of experimental results. When data is redundantly available from multiple sources, provenance may also be beneficial for automated data curation and arbitration of data inconsistencies.

D. Non-textual data

Biological data are not always textual. This adds to the difficulties of indexing data for effective data-mining. This is most notably the case for high throughput microscopy imaging and enzymatic activity experimental characterization. For example, microplate assays are widely used in research and drug discovery to detect biological or chemical events of samples. Those events are typically detected

by measuring the fluorescence intensity of samples from each of the ninety-six wells that compose a plate and are usually stored as greyscale pictures. SDS-PAGE (Sodium Dodecyl Sulfate PolyAcrylamide Gel Electrophoresis) gels are broadly used for protein separation and analysis. Gels are usually stored as images and analysed using imaging tools such as ImageJ [46] and to date, indexing and mining those files is not possible using current databases.

E. Data versioning

Data versioning offers many benefits to end users. First, it enables data recovery in case of an entry mistake or system corruption. More importantly, it facilitates the analysis of historical changes of data, which can be helpful to enhance predictions and automatic data classification [47]. Barkstrom presented a formal structure for keeping track of files, source code, scripts, and related material for large-scale Earth science data production [48]. However, biological data present a number of unique challenges and effective solutions for biological data versioning are still lacking.

One of the main issues in biological data versioning is the variability of the data and the lack of well-defined protocol to compare the numerous formats. Biological experiments are conducted using some biological data as input and yield results. These results may be used as data sources in other experiments to yield further sets of results. This cycle of experimentation continues, presumably until the required set of results are achieved. Research questions and interests may also vary over time, thereby altering the nature of the data. This is often the case for emerging high throughput technologies, where refinement of experimental protocols can significantly change the types of data to be collected [2].

IV. DATA WAREHOUSING FOR SYSTEMS BIOLOGY

To date the National Center for Biotechnology Information lists over 2,500 whole-genome sequencing projects, including 833 in progress. JGI lists nearly 1,500 whole-genome in addition to over 850 sequencing projects. Obtaining protein functional assignments is a necessary first step to enable progress in research areas associated with development, evolution and physiology. Solving the challenging problem of assigning a function to proteins requires multiple approaches because sequence similarity techniques may provide functional information for at most 50% of these proteins [49], [50]. Addressing this complex biological issue necessitates a *Systems Biology* [51], [52] approach to data analysis that requires identifying relationships hidden within multiple databases [53]. An important mechanism to achieve this goal will require the development of next-generation databases that enable sophisticated queries beyond simple text-based searches.

A. Systems biology: a new scientific perspective

Biological systems are more than a set of independent components working together. Although they are composed

of a limited number of elements, these elements — proteins in particular — usually have multiple functions and interact very tightly to form complex pathways. Historically, scientists have focused their research on isolating those elements to understand their individual functions and activities. The knowledge of their function is indeed critical to comprehend the intricate biological machinery of the whole system.

However, the success of the Human Genome Project has ushered in a new scientific perspective, a system-wide view of protein function and biological activity: while traditional methods focus on the detailed analysis of isolated proteins to understand its cellular function, systems biology applies a holistic approach to understand the details of cell biology and evolution as a whole. This relatively new concept in biology, which is expected to yield more realistic models of a complete biological system, requires the integration of data from the individual subsystems. Biological models are usually constructed — and validated — using high-throughput quantitative data including, but not limited to, genome sequencing, gene expression, proteomics, metabolomics and high-resolution microscopy imaging.

Effective integration of biological data should enable scientists to perform comparative analyses, modelling and inference of protein functions. The success of systems biology is therefore contingent on the ability to integrate and utilize a wide variety of types of data and computational techniques to automatically predict and assign functional annotations of proteins. Systems biology databases are expected to expand upon the traditional sequence and structure approach because the primary method to assign a function to a protein of unknown function is to identify a relationship with a protein of known function. Additional protein associations may also be made through protein interaction networks, metabolic pathways, protein expression patterns or any number of relationships envisioned by a biologist. The key to this approach is moving the design focus from a fixed database structure defining precomputed relationships between elements to a fluid and flexible relational model that can be adapted to the biologist questions [54] without re-designing the underlying data structure.

B. Overview of Existing Frameworks and their Limitations

Over the past few years, a number of specialized data warehouses have been developed to accommodate the specificities of biological data and to address specific needs. For example, the e-Fungi database [55] integrates data from 36 fungal genomes and aims at facilitating the systematic comparative study of those genomes. GeWare [56] is a laboratory information management system, featuring tools for the integrated analysis of clinical data from large biomedical research studies.

In this section, we briefly describe the main capabilities and limitations of a few general data warehousing frameworks although it is beyond the scope of this paper to

provide a formal and comprehensive benchmark.

1) *BioMart Central Portal*: BioMart Central Portal [26] is a complete framework that provides tools to federate a variety of biological databases. These include major biomolecular sequence, pathway and annotation databases. Moreover, the web server features a unified user-friendly interface for mining data from various datasets. The web server also supports programmatic access through a Perl API as well as RESTful web services. Queries are defined as a set of successive filters to be applied to data. Queries are however limited to two datasets at once, hence limiting the scaling capabilities of BioMart. It is also not possible to edit or create new filters, restricting possible queries to the otherwise comprehensive set of filters defined by the developers of the framework.

2) *BioXRT*: The BioXRT framework [57] is designed to allow biologists to publish their data on the Internet with only minimal knowledge of database design and usage. It provides a highly flexible and extensible database structure as well as tools to import spreadsheets. However, flexibility is achieved to the detriment of data types as all pieces of data are defined as strings of characters. Consequently, queries are constrained to string matching and BioXRT does not support advanced data-mining tools for complex biological questions.

3) *InterMine*: FlyMine [58] is a data warehouse built upon InterMine that integrates and utilizes numerous biological data sources. It features parsers for integrating data from numerous biological formats and facilities for adding one's own data. It provides a web access to integrated data at a number of different levels, from simple browsing to construction of complex queries and it includes a user-friendly web interface that can be easily customised for the user's needs. The provenance of the integrated datasets is also tracked. However, InterMine does not provide tools for classifying or clustering data and queries may not include similarity functions to address annotation errors as discussed in Section III-C.

4) *Open Genome Resource (OGeR)*: Strepto-DB [59] and SYSTOMONAS [60] are databases for the comparative genome analysis of streptococci and pseudomonas respectively, and rely on the Open Genome Resource (OGeR) to achieve data integration of external resources. OGeR is an open source system for the storage, visualization and analysis of prokaryotic genome data. Genome sequences and annotations can be automatically downloaded from relevant databases and features cross-references to external databases. However, like other frameworks, OGeR does not provide tools for clustering and statistical analysis, nor does it provide advanced mining tools besides pairwise and multiple sequence alignment tools.

5) *PROFESS*: The PROtein Function, Evolution, Structure and Sequence (PROFESS) database [61] integrates numerous biological databases and aims at giving an overview

of biological systems by integrating protein annotations at different levels: function, evolution, structure and sequence. The primary means to query the database is the “PROFESSor”, a unified text field that mines data from any integrated database. PROFESS also provides clustering and aggregation tools for statistical analysis of large datasets and features a user-friendly modular web interface. However, like other systems, its query system is based on a predefined non-customizable set of filters and does not yet support similarity functions besides standard BLAST searches.

V. CONCLUSION

Although biological data present a number of unique specificities making them challenging to integrate, there is a growing need for effective integration of biological datasets to enable large scale and comparative analysis of the numerous genomes being sequenced. To date, no biological data warehouse meets all the requirements for effective integration of system-wide data. BioXRT offers a flexible and extensible database structure, BioMart provides advanced data-mining tools although they may not be extended by users. PROFESS features a flexible and modular user interface and tools for clustering and statistical analysis of large datasets. InterMine also features a customizable user interface and is helpful to track the provenance of data.

However, there now exists a variety of resources that may be helpful in accommodating data inaccuracies, such as approximate string matching or similarity-based algorithms that may be implemented within database management systems for the next generation of biological data warehouses.

FUNDING

This work was supported by the Cellulosic Biofuel Network, funded by Agriculture and Agri-Food Canada.

REFERENCES

- [1] G. R. Cochrane and M. Y. Galperin, “The 2010 Nucleic Acids Research Database Issue and online Database Collection: a community of data resources.” *Nucleic acids research*, vol. 38, no. Database issue, pp. D1–4, Jan. 2010.
- [2] 3rd Millennium Inc., “Practical Data Integration in Biopharmaceutical R&D: Strategies and Technologies,” 2002.
- [3] R. Pottinger and A. Halevy, “MiniCon: A scalable algorithm for answering queries using views,” *The VLDB Journal*, vol. 10, no. 2-3, pp. 182–198, 2001.
- [4] A. Halevy, “Answering queries using views: A survey,” *VLDB Journal*, vol. 10, no. 4, pp. 270–294, 2001.
- [5] F. Giunchiglia, M. Yatskevich, and P. Shvaiko, “Semantic Matching: Algorithms and Implementation,” *Journal on Data Semantics*, vol. 9, pp. 1–38, 2007.
- [6] S. M. Falconer, N. F. Noy, and M.-A. Storey, “Ontology Mapping - a User Survey,” in *Proceedings of the 2nd International Workshop on Ontology Matching*. CEUR-WS.org, 2007, pp. 113–125.

- [7] C. Hewitt, "ORGs for Scalable, Robust, Privacy-Friendly Client Cloud Computing," *IEEE Internet Computing*, vol. 12, no. 5, pp. 96–99, Sep. 2008.
- [8] P. A. Babu, J. Udyama, R. K. Kumar, R. Boddepalli, D. S. Mangala, and G. N. Rao, "DoD2007: 1082 molecular biology databases." *Bioinformation*, vol. 2, no. 2, pp. 64–67, 2007.
- [9] H. Camacho, A. Cintado, and M. Duenas, "Technology evolution for genomic revolution," *Biotechnologia Aplicada*, vol. 22, no. 2, pp. 83–90, 2005.
- [10] W. Yang, H. Steen, and M. R. Freeman, "Proteomic approaches to the analysis of multiprotein signaling complexes." *Proteomics*, vol. 8, no. 4, pp. 832–851, Feb. 2008.
- [11] E. M. Lenz and I. D. Wilson, "Analytical strategies in metabolomics." *Journal of proteome research*, vol. 6, no. 2, pp. 443–458, Feb. 2007.
- [12] F. von Delft, D. McRee, and C. Kang, *Prospects for high-throughput structure determination by X-ray crystallography*. CRC Press, 2003, pp. 55–94.
- [13] K. D. Pruitt, T. Tatusova, W. Klimke, and D. R. Maglott, "NCBI Reference Sequences: current status, policy and new initiatives." *Nucleic acids research*, vol. 37, no. Database issue, pp. D32–36, Jan. 2009.
- [14] U.S. Department of Energy Joint Genome Institute, "Sequencing project," Nov. 2010. [Online]. Available: <http://www.jgi.doe.gov/sequencing/>
- [15] The Gene Ontology Consortium, "The Gene Ontology in 2010: extensions and refinements." *Nucleic acids research*, vol. 38, no. Database issue, pp. D331–335, Jan. 2010.
- [16] J. Muller, D. Szklarczyk, P. Julien, I. Letunic, A. Roth, M. Kuhn, S. Powell, C. von Mering, T. Doerks, L. J. Jensen, and P. Bork, "eggNOG v2.0: extending the evolutionary genealogy of genes with enhanced non-supervised orthologous groups, species and functional annotations." *Nucleic acids research*, vol. 38, no. Database issue, pp. D190–195, Jan. 2010.
- [17] M. Kanehisa, M. Araki, S. Goto, M. Hattori, M. Hirakawa, M. Itoh, T. Katayama, S. Kawashima, S. Okuda, T. Tokimatsu, and Y. Yamanishi, "KEGG for linking genomes to life and the environment," *Nucleic Acids Res*, vol. 36, no. Database issue, pp. D480–484, Jan. 2008.
- [18] L. Wong, "Technologies for integrating biological data," *Brief Bioinform*, vol. 3, no. 4, pp. 389–404, Dec. 2002.
- [19] D. I. Cooke-Fox, G. H. Kirby, and J. D. Rayner, "Computer translation of IUPAC systematic organic chemical nomenclature. 1. Introduction and background to a grammar-based approach," *Journal of Chemical Information and Modeling*, vol. 29, no. 2, pp. 101–105, May 1989.
- [20] D. Weininger, "SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules," *Journal of Chemical Information and Modeling*, vol. 28, no. 1, pp. 31–36, Feb. 1988.
- [21] A. McNaught, "The IUPAC international chemical identifier : InChI-A new standard for molecular informatics," *Chemistry international*, vol. 28, no. 6, pp. 12–14, 2006.
- [22] W. Fujibuchi, S. Goto, H. Migimatsu, I. Uchiyama, A. Ogihara, Y. Akiyama, and M. Kanehisa, "DBGET/LinkDB: an integrated database retrieval system." in *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, Jan. 1998, pp. 683–694.
- [23] U.S. Department of Energy Joint Genome Institute, "Genome database," Nov. 2010. [Online]. Available: <http://genome.jgi-psf.org/programs/fungi/>
- [24] P. J. Kersey, D. Lawson, E. Birney, P. S. Derwent, M. Haimel, J. Herrero, S. Keenan, A. Kerhornou, G. Koscielny, A. Kähäri, R. J. Kinsella, E. Kulesha, U. Maheswari, K. Megy, M. Nuhn, G. Proctor, D. Staines, F. Valentini, A. J. Vilella, and A. Yates, "Ensembl Genomes: extending Ensembl across the taxonomic space." *Nucleic acids research*, vol. 38, no. Database issue, pp. D563–569, Jan. 2010.
- [25] L. D. Stein, C. Mungall, S. Shu, M. Caudy, M. Mangone, A. Day, E. Nickerson, J. E. Stajich, T. W. Harris, A. Arva, and S. Lewis, "The generic genome browser: a building block for a model organism system database." *Genome research*, vol. 12, no. 10, pp. 1599–1610, Oct. 2002.
- [26] S. Haider, B. Ballester, D. Smedley, J. Zhang, P. Rice, and A. Kasprzyk, "BioMart Central Portal—unified access to biological data." *Nucleic acids research*, vol. 37, no. Web Server issue, pp. W23–27, Jul. 2009.
- [27] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *J Mol Biol*, vol. 215, no. 3, pp. 403–410, Oct. 1990.
- [28] C. Hadley, "Righting the wrongs," *EMBO Reports*, vol. 4, no. 9, pp. 829–831, 2003.
- [29] CODATA Task Group on biological macromolecules and colleagues, "Quality control in databanks for molecular biology," *BioEssays*, vol. 22, no. 11, pp. 1024–1034, Oct. 2000.
- [30] C. Jones, A. Brown, and U. Baumann, "Estimating the annotation error rate of curated GO database sequence annotations," *BMC Bioinformatics*, vol. 8, no. 1, 2007.
- [31] G. Navarro, "A Guided Tour to Approximate String Matching," *ACM Computing Surveys*, vol. 33, 1999.
- [32] J. B. Cushing, "Metadata and semantics: a computational challenge for molecular biology." *Omics : a journal of integrative biology*, vol. 7, no. 1, pp. 23–24, Jan. 2003.
- [33] J. A. Fox, S. McMillan, and B. F. F. Ouellette, "Conducting research on the web: 2007 update for the bioinformatics links directory." *Nucleic acids research*, vol. 35, no. Web Server issue, pp. W3–5, Jul. 2007.
- [34] W. R. Pearson and D. J. Lipman, "Improved tools for biological sequence comparison," *Proc Natl Acad Sci U S A*, vol. 85, no. 8, pp. 2444–2448, Apr. 1988.

- [35] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs," *Nucleic Acids Res*, vol. 25, no. 17, pp. 3389–3402, Sep. 1997.
- [36] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison, *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1999.
- [37] S. Dietmann, J. Park, C. Notredame, A. Heger, M. Lappe, and L. Holm, "A fully automatic evolutionary classification of protein folds: Dali Domain Dictionary version 3." *Nucleic acids research*, vol. 29, no. 1, pp. 55–57, Jan. 2001.
- [38] M. Levitt, "A unified statistical framework for sequence comparison and structure comparison," *Proceedings of the National Academy of Sciences*, vol. 95, no. 11, pp. 5913–5920, May 1998.
- [39] F. Armougom, S. Moretti, O. Poirot, S. Audic, P. Dumas, B. Schaeli, V. Kedua, and C. Notredame, "Expresso: automatic incorporation of structural information in multiple sequence alignments using 3D-Coffee." *Nucleic acids research*, vol. 34, no. Web Server issue, pp. W604–608, Jul. 2006.
- [40] D. J. States and D. Botstein, "Molecular sequence accuracy and the analysis of protein coding regions." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 88, no. 13, pp. 5518–5522, Jul. 1991.
- [41] Q. Ji, P. Haase, and G. Qi, "Combination of Similarity Measures in Ontology Matching using the OWA Operator," in *Proceedings of the 12th International Conference on Information Processing and Management of Uncertainty in Knowledge-Base Systems*. In: Proceedings of the 12th International Conference on Information Processing and Management of Uncertainty in Knowledge-Base Systems, 2008.
- [42] F. Damerau, "A technique for computer detection and correction of spelling errors," *Communications of the ACM*, vol. 7, no. 3, pp. 171–176, 1964.
- [43] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics Doklady*, 1966.
- [44] P. Buneman, S. Khanna, and W. C. Tan, "Data Provenance: Some Basic Issues," *Lecture Notes In Computer Science*; Vol. 1974, p. 87, 2000.
- [45] C. Goble, "Position Statement: Musings on Provenance, Workflow and (Semantic Web) Annotations for Bioinformatics," in *Workshop on Data Derivation and Provenance*, Chicago, USA, 2002.
- [46] W. Rasband, "ImageJ," Nov. 2010. [Online]. Available: <http://imagej.nih.gov/ij/>
- [47] P. Revesz and T. Triplet, "Temporal Data Classification Using Linear Classifiers," *Information Systems*, vol. 36, no. 1, pp. 30–41, 2011.
- [48] B. Barkstrom, *Data Product Configuration Management and Versioning in Large-Scale Production of Satellite Scientific Data*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2003, vol. 2649, pp. 118–133.
- [49] Y. Pouliot, J. Gao, Q. J. Su, G. G. Liu, and X. B. Ling, "DIAN: a novel algorithm for genome ontological classification." *Genome research*, vol. 11, no. 10, pp. 1766–1779, Oct. 2001.
- [50] L. J. Jensen, R. Gupta, H.-H. Staerfeldt, and S. Brunak, "Prediction of human protein function according to Gene Ontology categories." *Bioinformatics (Oxford, England)*, vol. 19, no. 5, pp. 635–642, Mar. 2003.
- [51] T. Ideker, T. Galitski, and L. Hood, "A new approach to decoding life: systems biology." *Annual review of genomics and human genetics*, vol. 2, pp. 343–372, Jan. 2001.
- [52] H. Kitano, "Systems biology: a brief overview." *Science*, vol. 295, no. 5560, pp. 1662–1664, Mar. 2002.
- [53] A. R. Joyce and B. O. Palsson, "The model organism as a system: integrating 'omics' data sets," *Nat Rev Mol Cell Biol*, vol. 7, no. 3, pp. 198–210, Mar. 2006.
- [54] R. Stevens, "A classification of tasks in bioinformatics," *Bioinformatics*, vol. 17, no. 2, pp. 180–188, Feb. 2001.
- [55] C. Hedeler, H. M. Wong, M. J. Cornell, I. Alam, D. M. Soanes, M. Rattray, S. J. Hubbard, N. J. Talbot, S. G. Oliver, and N. W. Paton, "e-Fungi: a data resource for comparative analysis of fungal genomes." *BMC genomics*, vol. 8, no. 1, p. 426, Jan. 2007.
- [56] E. Rahm, T. Kirsten, and J. Lange, "The GeWare data warehouse platform for the analysis of molecular-biological and clinical data," *Journal of Integrative Bioinformatics*, vol. 4, no. 1, 2007.
- [57] J. Zhang, G. E. Duggan, R. Khaja, and S. W. Scherer, "BioXRT: a novel platform for developing online biological databases based on the Cross-Referenced Tables model," in *3rd Canadian Working Conference on Computational Biology*, Markham, Canada, 2004.
- [58] R. Lyne, R. Smith, K. Rutherford, M. Wakeling, A. Varley, F. Guillier, H. Janssens, W. Ji, P. McLaren, P. North, D. Rana, T. Riley, J. Sullivan, X. Watkins, M. Woodbridge, K. Lilley, S. Russell, M. Ashburner, K. Mizuguchi, and G. Micklem, "FlyMine: an integrated database for *Drosophila* and *Anopheles* genomics." *Genome biology*, vol. 8, no. 7, p. R129, Jan. 2007.
- [59] J. Klein, R. Münch, I. Biegler, I. Haddad, I. Retter, and D. Jahn, "Strepto-DB, a database for comparative genomics of group A (GAS) and B (GBS) streptococci, implemented with the novel database platform 'Open Genome Resource' (OGer)." *Nucleic acids research*, vol. 37, no. Database issue, pp. D494–8, Jan. 2009.
- [60] C. Choi, R. Munch, B. Bunk, J. Barthelmes, C. Ebeling, D. Schomburg, M. Schobert, and D. Jahn, "Combination of a data warehouse concept with web services for the establishment of the *Pseudomonas* systems biology database SYSTOMONAS," *Journal of Integrative Bioinformatics*, vol. 4, no. 1, 2007.
- [61] T. Triplet, M. Shortridge, M. Griep, J. Stark, R. Powers, and P. Revesz, "PROFESS: a PROtein Function, Evolution, Structure and Sequence database." *Database : the journal of biological databases and curation*, p. baq011, Jan. 2010.

Studying the Impact of Partition on Data Reduction for Very Large Spatio-temporal Datasets

Nhien An Le Khac

School of Computer Science
University College Dublin
Belfield, Dublin 4, Ireland
e-mail: an.lekhac@ucd.ie

Martin Bue

Ecole Polytechnique Universitaire de
Lille
Villeneuve d'Ascq cedex, France
e-mail: Martin.Bue@polytech-lille.net

M-Tahar Kechadi

School of Computer Science
University College Dublin
Belfield, Dublin 4, Ireland
e-mail: tahar.kechadi@ucd.ie

Abstract—Nowadays, huge amounts of data are being collected with spatial and temporal components from sources such as metrological, satellite imagery etc. Efficient visualisation as well as discovery of useful knowledge from these datasets is therefore very challenging and becoming a massive economic need. Data Mining has emerged as the technology to discover hidden knowledge in very large amounts of data. Furthermore, data mining techniques could be applied to decrease the large size of raw data by retrieving its useful knowledge as representatives. As a consequence, instead of dealing with a large size of raw data, we can use these representatives to visualise or to analyse without losing important information. Recently, we proposed a new approach based on different clustering techniques for data reduction to help analyse large spatio-temporal data. This approach is based on the partition of huge datasets due to the memory constraint. In this paper, we evaluate the impact of various numbers of partitions on our data reduction approach.

Keywords-spatio-temporal datasets; data reduction; data partition; density-based clustering; shared nearest neighbours

I. INTRODUCTION

Many natural phenomena present intrinsic spatial and temporal characteristics. Besides traditional applications, recent concerns about climate change, the threat of pandemic diseases, and the monitoring of terrorist movements are some of the newest reasons why the analysis of spatio-temporal data has attracted increasing interest. With the recent advances in hardware, high-resolution spatio-temporal datasets are collected and stored to study important changes over time, and patterns of specific events. However, these datasets are often very large and grow at a rapid rate. So, it becomes important to be able to analyse, discover new patterns and trends, and display the results in an efficient and effective way.

Spatio-temporal datasets are often very large and difficult to analyse [1][2][3]. Fundamentally, visualisation techniques are widely recognised to be powerful in analysing these datasets [4], since they take advantage of human abilities to perceive visual patterns and to interpret them [5]. However, spatial visualisation techniques currently provided in the existing geographical applications are not adequate for decision-support systems when used alone. For instance, the problems of how to visualise the spatio-temporal multi-

dimensional datasets and how to define effective visual interfaces for viewing and manipulating the geometrical components of the spatial data [6] are the challenges. Hence, alternative solutions have to be defined. Indeed, new solutions should not only include a static graphical view of the results produced during the data mining (DM) process, but also the possibility to obtain dynamically and interactively different spatial and temporal views as well as interact in different ways with them. DM techniques have been proven to be of significant value for analysing spatio-temporal datasets [7][8]. It is a user-centric, interactive process, where DM experts and domain experts work closely together to gain insight on a given problem. In particular, spatio-temporal data mining is an emerging research area, encompassing a set of exploratory, computational and interactive approaches for analysing very large spatial and spatio-temporal datasets. However, several open issues have been identified ranging from the definition of techniques capable of dealing with the huge amounts of spatio-temporal datasets to the development of effective methods for interpreting and presenting the final results.

Analysing a database of even a few gigabytes is an arduous task for machine learning techniques and requires advanced parallel hardware and algorithms. Huge datasets create combinatorially explosive search spaces for DM algorithms which may make the process of extracting useful knowledge infeasible owing to space and time constraints. An approach for dealing with the intractable problem of learning from huge databases is to select a small subset of data for mining [2]. It would be convenient if large databases could be replaced by a small subset of representative patterns so that the accuracy of estimates (e.g., of probability density, dependencies, class boundaries) obtained from such a reduced set should be comparable to that obtained using the entire dataset. This approach can be viewed as similar to sampling, a technique that is commonly used for selecting a subset of data objects to be analysed. There are different techniques for this approach such as the scaling by factor [9][10], data compression [11], clustering [12], etc. Recently, we proposed a reduction technique based on clustering [13] for large size of spatio-temporal datasets. Clustering is used on spatio-temporal data to take advantage of the fact that, objects that are close together in space and/or in time can usually be grouped together. As a consequence, instead of dealing with a large size of raw data, we can use these cluster

representatives to visualise or to analyse without losing important information [13]. In this solution, multi-partition approach has been applied to cope with huge size of input datasets i.e., all draw dataset is divided in equal parts and each time, only one partition can be processed. Basically, the number of partitions depends on the runtime memory. We normally optimise the number of partitions by maximise the size of each partition in according to the size of memory. Indeed, in order to exploit efficiently the capacity of multithreading on the multi-core platforms, a larger number of partitions in smaller sizes is required. However, the changing of partition size could effect on final representatives because of changing the position of nearest neighbours. A best trade-off between the number of partitions and final representatives should be determined. To do so, in this paper, we study the impact of the number of partitions on final representative results.

The rest of this paper is organised as follows. In Section II, we present background related to this subject. Section III describes briefly our data reduction technique based on clustering techniques. We discuss on the impact of the number of partitions in Section IV and we evaluate then this impact on real large spatio-temporal dataset in Section V. Section VI is to deal with the conclusion and our future work.

II. BACKGROUND

In this section, we present firstly the spatio-temporal data mining system and then different clustering techniques applied for reducing spatio-temporal datasets.

A. Spatio-temporal data mining

Spatio-temporal DM represents the junction of several research areas including machine learning, information theory, statistics, databases, and geographic visualisation. It includes a set of exploratory, computational and interactive approaches for analysing very large spatial and spatio-temporal datasets. Recently various projects have been initiated in this area ranging from formal models [4][14] to the study of the spatio-temporal data mining applications [5][14]. In spatio-temporal data mining the two dimensions “spatial” and “temporal” have added substantial complexity to the traditional DM process. It is worth noting, while the modelling of spatio-temporal data at different levels of details presents many advantages for both the application and the system. However it is still a challenging problem. Some research has been conducted to integrate the automatic zooming of spatial information and the development of multi-representation spatio-temporal systems [15][16][17]. However, the huge size of datasets is an issue with these approaches.

In [8][9], the authors proposed a strategy that is to be incorporated in a system of exploratory spatio-temporal data mining, to improve its performance on very large spatio-temporal datasets. This system provides a DM engine that can integrate different DM algorithms and two complementary 3-D visualisation tools. This approach reduces their datasets by scaling them by a factor F; it simply runs through the whole dataset taking one average value for

the F^3 points inside each cube of edge F. This reducing technique has been found to be inefficient as a data reduction method which may lose a lot of important information contained in the raw data.

B. Data reduction by clustering

To the best of our knowledge, there is only our recently work [12][13] which proposed a clustering-based data reduction method in the context of analysing spatio-temporal datasets. This method is based on clustering [1] to cope with the huge size of spatio-temporal datasets in order to facilitate the mining of these datasets. The main idea is to reduce the size of that data by producing a smaller representation of the dataset, as opposed to compressing the data and then uncompressing it later for reuse. The reason is that we want to reduce and transform the data so that it can be managed and mined interactively. Clustering technique used in this approach is K-Medoids [12] and density-based [18]. The advantage of these techniques is simple in term of basic algorithms used. In the first technique, its representatives (medoids points) cannot however reflect adequately all important features of the datasets because it is not sensitive to the shape of the datasets (convex) as the second technique does with its specific core points [20] used as representatives. A brief presentation of this second technique is in the following section.

III. DENSITY-BASED DATA REDUCTION

As described in [8][9], our spatio-temporal data mining framework consists of three steps: data preparation, data mining and visualisation. The visualisation step contains different visualisation tools that provide complementary functionality to visualise and interpret mined results. Normally, the raw spatial-temporal dataset is too large for any algorithm to process; the goal of the data preparation step is to reduce the size of that data by producing a smaller representation of the dataset, so-called representatives, without losing any relevant information, as opposed to compressing the data and then uncompressing it later for reuse. Furthermore, we aim to reduce and transform the data so that it can be managed and mined interactively. It allows the mining step to apply mining technique such as clustering, association rules on the representatives (tightly grouped data objects) to produce new knowledge and ready for evaluation and interpretation.

We proposed a new data reduction method based on clustering to help with the mining of the very large spatio-temporal dataset [13]. The use of cluster representatives helps to filter (reduce) datasets without losing important/interesting information because clustering techniques group data objects based on the characteristics of the objects and their relationships. It aims at maximising the similarity within a group of objects and the dissimilarity between the groups in order to identify interesting structures in the underlying data. Concretely, we have implemented a combination of a density-based clustering - a modification of DBSCAN algorithm [18]; and a graph-based clustering - a Shared Nearest Neighbor Similarity (SNN) algorithm [19], in this approach. The advantage of this combination has been

discussed in [13]. Briefly, this combination is not only efficient with spatial datasets as it takes into account the shape (convex) of the data points but also addresses the problems of low similarity and differences in density. The whole can be resumed as follow: (1) a pre-processing is applied on raw datasets to filter NULL values. (2) SNN similarity graph is built for all datasets. Similarity degree of each data object is also computed in this step. The two parameters Eps and $Minpts$ are selected based on these similarity degrees. Next, a DBSCAN-based algorithm is carried out on the datasets to determine *core objects*, *specific core objects*, *density-reachable objects* and *density-connected objects*. The definition of these objects are defined in [18][20]. *Core objects* or *specific core objects* are selected as cluster representatives that form a new (meta-) dataset (3). This dataset can then be analysed and produce useful information (i.e., models, rules, etc.) by applying other DM techniques (the mining step). It is important to note that data objects that have a very high similarity between each other can be grouped together in the same clusters. As a result of this step, the new dataset is much smaller than the original data without losing any important information from the data that could have an adverse effect on the result obtained from mining the data at a later stage. Fig. 1 and Fig. 2 show respectively all data points before and after this density-based clustering approach on around 25 million data points of 4 dimensions X, Y, Z, QCLOUD for the time step 2 of the Isabel hurricane datasets [21]. The representatives (Fig. 2) could reflect the general shape of hurricane based on (X,Y,Z,QCLOUD) comparing to their whole original points (Fig.2). More results and discussion for other time-steps can be found in [13].

Memory issue. The sizes of spatio-temporal datasets are normally very large. For instance, Hurricane Isabel dataset [21] is represented by a space of $500 \times 500 \times 100$ with 25×10^6 data points (data objects). Therefore, a dissimilarity matrix (used to determine the distance between any two data points in the dataset) of all data points would take $25^2 \times 10^{12} \times 6$ bytes. It exceeds the memory capacity of commodity computation machines. Normally, tree-based topologies such as R-tree [22] have been applied to index data points in order to tackle this memory issue. However, high frequency of queries on a huge tree structure and of secondary memory access would be a performance impact. By using SNN algorithm in our approach, we only need to keep the distances to k-nearest neighbour of each data point. For instance, for Hurricane Isabel dataset presented above, we would need $25 \times 10^6 \times 6 \times k$ bytes of memory for storing k-nearest information of all data points.

IV. STUDYING THE IMPACT OF THE NUMBER OF PARTITIONS ON REDUCTION RESULTS

In our approach, if the computational variables of the whole space cannot be all loaded into main memory, a local-global solution will be applied. In this solution, a whole dataset is equally divided in un-joint partitions. The number of partition depends on the memory size where one partition can be processed at each time. Let D , P_i , M , S_p be all raw datasets, partition i , memory reserved for this computation

and computational memory needed for each partition respectively, this problem can be officially defined as:

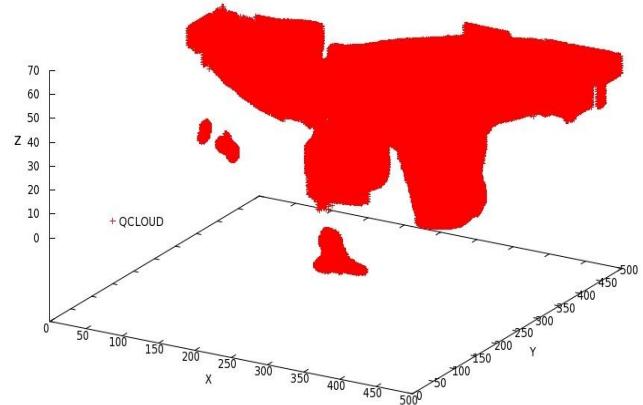


Figure 1. All datasets for QCLOUD, timestep 2.

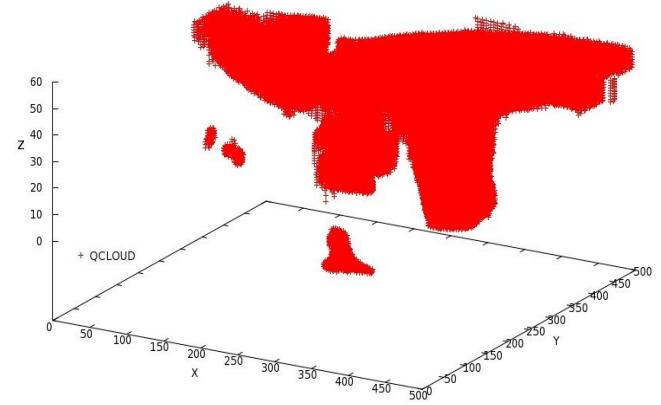


Figure 2. Data reduction by SNN-DBSCAN for QCLOUD, timestep 2.

$$D = \bigcup_{i=1}^n P_i, \forall i, j \in \{1..n\}, i \neq j : P_i \cap P_j = \emptyset \quad (1)$$

$$M = S_p \quad (2)$$

For each partition P_i , we apply our algorithm described above to determine *core objects* C_{P_i} , *specific core objects* SC_{P_i} and *clusters* Cl_{P_i} . As mentioned above, both *core objects* and *specific core objects* can be used as representatives. However, according to the experiments [13], the ratio between the number of core objects and the number of all raw data objects is relative high (~90%). Thus, only *specific core objects* are considered *local representatives* and clusters Cl_{P_i} are called *local clusters*. After processing all partitions, there are two methods to build the *global clusters*:

- merging local clusters together based on their core objects and the distance ε between core objects located on the borders of difference partitions to build global clusters and the global representatives (union of all local representatives). This method is

simple. However, the determination of ε is an issue because of the variety in density of different partitions. Indeed, our reduction approach is based on SNN degree [19], not on the distance.

- all local representatives will be joined together to create a new dataset. A DBSCAN-based algorithm will be carried out on this new dataset to create new clusters called *global clusters*. The *core objects* are *global representatives*. Our studying is focusing on this method.

The *global representatives* of *global clusters* form a reduction dataset. In both methods above, the number of partition n is an important impact on the final reduction results. When we increase n , the number of local representatives is also increased and that affects to the global results, especially the ones locate near the border of each partition because most of them would not be representatives at the global level. Basically, the smaller number n is expected in according to the constraint of the running memory and in the optimal case $M \equiv S_{P_i}$ (cf. (2)) i.e., only one partition is loaded with its maximum size in according to the computational memory reserved.

Meanwhile, today, running efficiently applications on multi-core platforms is a challenge as the multi-core hardware is widely used; the software has not been ready yet. In the context of analysing huge size of spatio-temporal datasets, the multi-core programming is a solution for improving the processing time. To do so, these current approaches should be designed in multicore-ready style i.e., it allows to exploit the maximum capacity of multithreading on the target system. Concretely, in our approach, multithreading can be applied in two different methods. In the first one, multithreading is carried out on one partition P_i loaded in the memory. In this case, important changes are needed for the algorithm in according to multithreading environment. In the second method, multi partitions can be loaded in the memory. This case can efficiently benefit the capacity of multithreading without changing the algorithm. In this case:

$$M \equiv \bigcup S_{P_j}, P_j \in D, \text{Count}(S_{P_j}) = m \quad (3)$$

However, increasing the number of partitions would lead to a performance issue in term of final reduction results as discussed above. So, we should determine the optimal m i.e., the value maximum of m that does not affect the final results. On the other hand, a large number of partitions is also a running time issue as most of processing time is reserved for loading/unloading datasets from/to computational memory instead of calculating.

V. EVALUATION

In this section, we study the impact m (c.f. (3)) with real spatio-temporal datasets in the context of data reducing by using DM techniques described in Section III. The dataset is the Isabel hurricane data [21] produced by the US National Centre for Atmospheric Research (NCAR). It covers a period of 48 hours (time-steps). Each time-step

contains several atmospheric variables. The grid resolution is $500 \times 500 \times 100$. The total size of all files is more than 60GB (~ 1.25 GB for each time-step). The experimentation details and a discussion are given below.

The platform of our experimentation is a PC of 3.4 GHz Dual Core CPU, 1GB RAM using Java 1.6 on Linux kernel 2.6. Datasets of each time-step include 13 non-spatio attributes, so-called dimensions. In this evaluation, QCLOUD is chosen for analysis; it is the weight of the cloud water measured at each point of the grid. The range of QCLOUD value is $[0 \dots 0.00332]$. The chosen time-step is 2 as the different time steps is similar in term of processing. We also filter the NULL value and land value of testing dataset. Totally, the testing dataset contains around 25 million data points of 4 dimensions X, Y, Z, QCLOUD for each time step. After the reduction process, the number of data objects is around 100000. Due to the memory constraint, the number of partitions is varied from 40 to 180.

Figures from 3 to 7 show the clustering results for 40, 70, 100, 120 and 180 partitions respectively. In each figure, we only show 9 biggest clusters with different colours¹. Other clusters are in the same colour: black-(9). Table 1 gives more details on the number of clusters, the number of representatives (specific core objects) for each case of partition. By observing these figures, we recognise that:

- the three biggest clusters (blue-(0), red-(1) and green-(2)) are similar in all cases i.e., they are not strongly affected by the number of partitions. This also means that if the size of a cluster is large enough it then can preserve its shape against a reasonable number of fragmentations. The optimal relationship between the size of cluster and this number is out of scope of this paper.
- small difference in the number of clusters and their shapes among cases of 40, 70, 100 and 120 partitions. In the 40-partition case (Fig. 3), the 9th cluster (grey (8)) is clearer than the rest. The reason is that it gets a smaller fragmentation degree of raw data than other cases.
- The three cases 70, 100 and 120-partition are similar in terms of cluster shape and cluster position.
- Size and shape of clusters from the 4th one (yellow-(3)) of the 180-partition case is different compared to other cases. This means that this number of partitions affects on the final results.

Besides, as shown in the Table 1, the number of clusters created by all cases is similar with the difference is less than 5% (varied from 82 clusters to 87 clusters). Indeed, the number of representatives (specific core objects) is decreasing when we increase the number of partitions. Because there is an increasing of local representatives, that are not global representatives, in each partition. However this

¹ We use different colours rather than makers ('+', '*', 'x...') to distinguish different clusters because of the large number of plotting points (~ 100000). We put however the number (0, 1, 2...) on each cluster to make it easier to distinguish with other clusters in the case of black & white hardcopy.

difference among these numbers is quite small (less than 1%).

Basing on these observations, we can vary the number of partitions loaded in the computational memory at the same time to exploit efficiently the multithreading capacity offered by multi-core platforms. Concretely, as shown in this experiment, we should start at 40-partition case with one partition loaded due to the memory constraint; we can obviously load from 2 to 4 partitions at the same time with a minor effect on the final results.

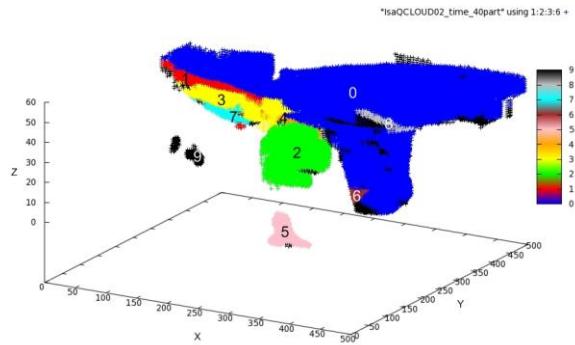


Figure 3. 40-partition.

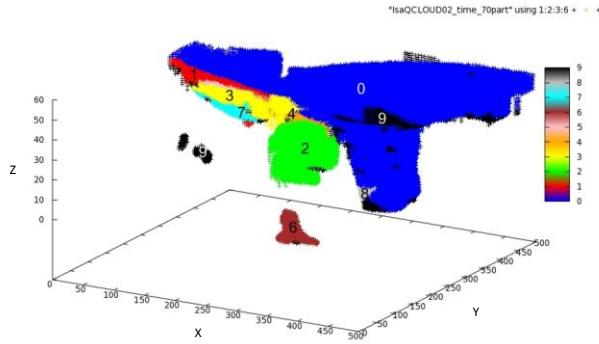


Figure 4. 70-partition.

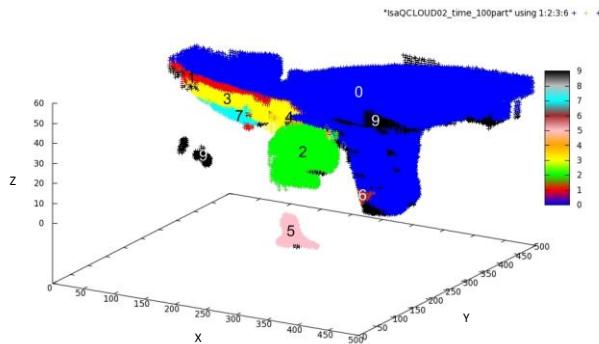


Figure 5. 100-partition.

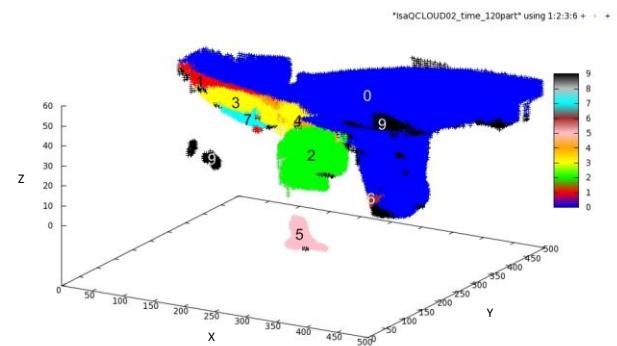


Figure 6. 120-partition.

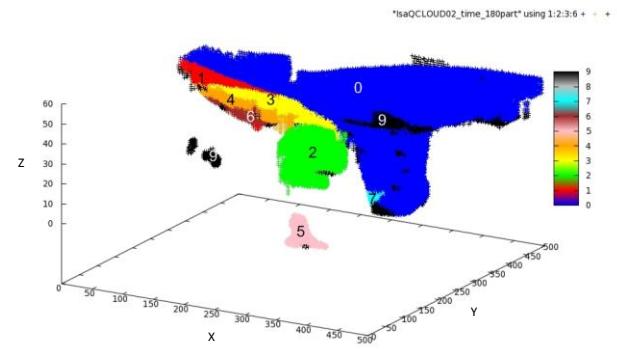


Figure 7. 180-partition.

As a brief conclusion, these experiments above show that we can process more partitions at the same time to benefit the multi-core environment in order to increase the speed-up of processing time.

TABLE I. REPRESENTATIVES AND CLUSTERS OF DIFFERENT NUMBER OF PARTITIONS

Number of partitions	Number of representatives	Number of clusters
40	103422	82
70	103139	84
100	102901	87
120	102708	86
180	102295	83

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we study the impact of the number of partitions on final representative results in the context of using clustering techniques for reducing the large size of spatio-temporal datasets. As there is a limitation of main memory, then the multi-partition approach has been applied.

Basically, a minimum number of partitions is expected to benefit the memory capacity as well as to preserve the final results. Besides, in order to exploit efficiently multi-core platforms, more partitions need to be processed at the same time. Thus, an optimal number of partitions should be determined. The experimental results for QCLOUD of the Isabel hurricane in its space X,Y,Z for one time-step show that we can increase the number of partition with a smaller size loaded in the computational memory without effecting the final results. In the case where we should tackle with huge size of datasets, if the speedup is increasing from 3 to 4 times then it is an important gain in term of processing time.

A more extensive evaluation is on-going. In the future we intend to analyse different combinations of dimensions over more time steps to determine an optimal number of partitions for all cases. Indeed, we also carry out tests with multithreading techniques on multi-core platforms to evaluate the speedup gain as well as to prove the robustness of our approach of reducing spatio-temporal datasets.

REFERENCES

- [1] Dunham, M. H., Data Mining: Introductory and Advanced Topics, Prentice Hall, 2003
- [2] Tan, P-N., Steinbach, M. and Kumar, V., Introduction to Data Mining, Addison Wesley, 2006
- [3] Ye, N. (ed), The Handbook of Data Mining. Lawrence Erlbaum Associates Publishers, Mahwah, New Jersey, USA 2003
- [4] Johnston W.L., "Model visualisation, in: Information Visualisation in Data Mining and Knowledge Discovery", Morgan Kaufmann, Los Altos, CA, 2001, pp. 223–227.
- [5] Andrienko N., Andrienko G., and Gatalsky P., "Exploratory Spatio-Temporal Visualisation: an Analytical Review", Journal of Visual Languages and Computing, special issue on Visual Data Mining, December, v.14 (6), 2003, pp. 503-541.
- [6] Liu, H. and Motoda, H., On Issues of Instance Selection, Data Mining Knowledge Discovery 6, 2, April, 2002, pp.115-130.
- [7] Roddick, J. F., Hornsby, K., and Spiliopoulou, M., An Updated Bibliography of Temporal, Spatial, and Spatio-temporal Data Mining Research. Proceedings of the First International Workshop on Temporal, Spatial, and Spatio-Temporal Data Mining-Revised Papers, September 12, 2000, pp.147-164.
- [8] Roddick, J.F. and Lees, B.G., Paradigms for Spatial and Spatio-Temporal Data Mining, In Geographic Data Mining and Knowledge Discovery. Miller H. and Han J. (Eds), Taylor & Francis, 2001
- [9] Compiepa, P., Di Martino, S., Bertolotto, M., Ferrucci, F., and Kechadi, T., Exploratory Spatio-Temporal Data Mining and Visualization. Journal of Visual Languages and Computing, 18, 3, June, 2007, pp.255-279.
- [10] Bertolotto, M., Di Martino, S., Ferrucci, F., and Kechadi, T., Towards a Framework for Mining and Analysing Spatio-Temporal Datasets, International Journal of Geographical Information Science, 21, 8, July, 2007, pp.895-906.
- [11] Sayood, K., Introduction to Data Compression, 2nd Ed., Morgan Kaufmann, 2000
- [12] Whelan, M., Le-Khac, N-A. and Kechadi, M-T., Data Reduction in Very Large Spatio-Temporal Data Sets, IEEE International Workshop On Cooperative Knowledge Discovery and Data Mining 2010 (WETICE 2010), Larissa, Greece, June 2010
- [13] Le-Khac, N-A., Bue, M., Whelan, M., and Kechadi, M-T., A clustering-based data reduction for very large spatio-temporal datasets, The 6th International Conference on Advanced Data Mining and Applications (ADMA2010), Springer Verlag LNAI, November 19-21, 2010, ChongQing, China (to appear)
- [14] Costabile M.F., Malerba D. (Editors), Special Issue on Visual Data Mining, Journal of Visual Languages and Computing, Vol. 14, December, 2003, pp.499-510.
- [15] Bettini, C., Dyleson, C.E., Evans, W.S., Snodgrass, R.T., A glossary of time granularity concepts. In Proceedings of Temporal Databases: Research and Practice, Lecture Notes in Computer Science, Vol. 1399, Springer-Verlag, 1998, pp. 406–413.
- [16] Bettini, C., Jajodia, S., Wang, X., Time Granularities in Databases, Data Mining, and Temporal Reasoning, Springer-Verlag, 2000
- [17] Cattell, R., et al., The Object Database Standard: ODMG 3.0, Morgan-Kaufmann, 1999
- [18] Ester, M., Kriegel, H.-P., Sander, J., Xu, X., A Density-Based Algorithm for Discovering clusters in Large Spatial Databases with Noise. Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD'96), pp.226-231, Portland, OR, USA, 1996
- [19] Jarvis, R. A. and Patrick, E.A., Clustering using a similarity Measure Based on shared Nearest Neighbours. IEEE Transactions on Computers, C-22(11), 1973, pp.1025-1034.
- [20] Januzaj, E., Kriegel, H-P., Pfeifle, M., DBDC: Density-Based Distributed Clustering. Proc. 9th Int. Conf. on Extending Database Technology (EDBT), Greece, 2004, pp.88-105.
- [21] National Hurricane Center, Tropical Cyclone Report: Hurricane Isabel, <http://www.tpc.ncep.noaa.gov/2003isabe.lshtml>, 2003
- [22] Guttman, A., R-Trees: A Dynamic Index Structure for Spatial Searching. Proc. ACM SIGMOD International Conference on Management of Data, 1984, pp. 47–57.

Hybrid DCT-CT Digital Image Adaptive Watermarking

Bijan Fadaenia, Nasim Zarei

Electrical engineering department

*Islamic Azad University-Hamedan Branch
Hamedan, Iran*

Fadaenia@iauh.ac.ir, Nasim.zarei@iauh.ac.ir

Abstract—This paper proposes a robust and blind digital image watermarking which uses a correlation based algorithm to embed a binary pseudo-random sequence into a grayscale host image. This scheme applies a combination of Discrete Cosine Transform (DCT) and Contourlet Transform (CT). Due to increasing the imperceptibility of embedded watermark, the power of watermark is varying in different regions of host image. The varying watermark power calculation is based on fuzzy decision. Experimental results show that the proposed method is robust against both geometric and non-geometric attacks.

Keywords-Watermarking; DCT; Contourlet; Fuzzy.

I. INTRODUCTION

Digital watermarking has been identified as a possible solution for Copyright protection of digital media and has become an area of increased research activity over the last decade [1]. Commonly, a digital watermark is a code that is embedded into a media. It plays the role of a digital signature, providing the media with a sense of ownership or authenticity. The primary benefit of watermarking is that the content is not separable from the watermark [2]. In the case of digital image this technique tries to embed invisible information in digital image. As mentioned before, the digital watermark must be robust against media manipulations [3].

Although most of works in the field of watermarking focuses on using the multiresolution analysis proposed by Wavelet Transform [4], [5], [6], [7], Contourlet Transform began to gain some interests for its capability of capturing directional information such as smooth contours, and directional edges [1], [8], [9], [10], [11], [12].

In this paper, we compute two level CT of host image then we divide selected sub-bands into blocks and apply DCT to each block. Next, we generate two pseudo-random uncorrelated sequences for embedding 0 and 1 and alter the CT–DCT coefficients by using a fuzzy system. We will show that the proposed algorithm can resist against both geometric and nongeometric attacks and increase the PSNR.

This paper is organized as follows: in Section 2, a quick view of CT and its advantages for watermarking will be provided. The proposed algorithm will be introduced in Section 3. Experimental results will be shown in Section 4. Finally, Section 5 concludes the paper.

II. CONTOURLET TRANSFORM (CT)

The Discrete Contourlet Transform is a relatively new transform which was proposed by Do et al. [13]. The main feature of this transform is the potential to efficiently handle 2-D singularities, unlike wavelets which can deal with point singularities exclusively. This difference is caused by two main properties that the CT possess: 1) The directionality property, as opposed to only 3 directions of wavelets. 2) The anisotropy property, meaning that the bases functions appear at various aspect ratios (depending on the scale), whereas wavelets are separable functions and thus their aspect ratio equals to 1. The main advantage of the CT over other geometrically-driven representations, e.g., curvelets [14] and bandelets [15], is its relatively simple and efficient wavelet-like implementation using iterative filter banks. Due to its structural resemblance with the wavelet transform (WT), many image processing tasks applied on wavelets can be seamlessly adapted to contourlets.

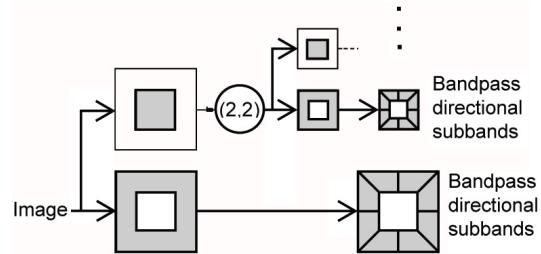


Figure 1: Contourlet Filter Bank

The CT is constructed by two filter-bank stages, a Laplacian Pyramid (LP) [16] followed by a Directional Filter Bank (DFB) [17] as shown in Fig. II. The LP decomposes the image into octave radial-like frequency bands to capture the point discontinuities, while the DFB decomposes each LP detail band into many directions (a power of 2) to link these point discontinuities into linear structures Fig. 2. In CT the HF subband is created by subtracting the G-filtered LF subband from the original image [1]. In this case, if we change the HF coefficients, the LF coefficients will be affected likely. Because of the characteristic of LP, the CT is evidently different from the WT. In the WT, the HF subband is created by filtering the original image with high-pass filter.

Therefore, the change of HF coefficients does not affect the LF coefficients. Because the WT does not have the spreading effect as the LP, the embedded watermark is susceptible to the attacks such as low-pass filtering, quantization and compression that destroy the HF coefficients of the image seriously. In contrast, if the watermark is embedded into the largest detail subbands of CT, it is likely to be spread out into all subbands when we reconstruct the watermarked image. Thus, the watermarking scheme in CT domain may be robust to the widely spectral attacks resulting from both LF image processing and HF image processing

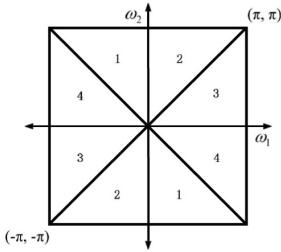


Figure 2: Frequency partitioning with four real wedge-shaped frequency bands.

III. PROPOSED ALGORITHM

In this paper, a 1225 bits pseudo-random sequence is used as watermark. In this algorithm sub-bands 2 and 3 in second level are selected (Fig. 2), because human eyes are less sensitive to noise in oblique orientation [18] and these subbands are the most oblique [13].

A. Watermark embedding

Step 1 : Two level contourlet is applied to image and image is divided into nine sub-bands.

Step 2 : Sub-bands 2 and 3 of second level are selected for watermark insertion.

Step 3 : Selected sub-bands are divided into $N \times N$ blocks and the DCT of each block is computed. These blocks are denoted as $block_i$, $i = 1, 2, \dots, M$ and $M \leq N_w$, which N_w is the number of watermark bits. If dimensions of watermark logo are denoted as L_w and H_w then $N_w = L_w \times H_w$. Table I, is shown that the combination of CT and DCT can increase the quality of watermarked image. Table II, is shown that the combination of CT and DCT can increase the robustness of algorithm against both geometric and non-geometric attacks by comparing NC values.

Step 4 : If each element of watermark is denoted as W_j , $j = 1, 2, \dots, N_w$. Two uncorrelated pseudo-random sequences with zero mean are generated, one of them is used for embedding $W_j = 0$ and the other one for $W_j = 1$, which denoted as PN_0 and PN_1 respectively.

Step 5 : To embed watermark into the produced blocks, the middle frequency coefficients of each block are selected and denoted as Y. This selection is a tradeoff between robustness and imperceptibility of watermark.

Step 6 : The optimum watermarking weight, α , is calculated for each block using a fuzzy system. The used fuzzy system will be explained later.

Step 7 : The watermark bits are embedded into image as follows:

$$Y' = Y + \alpha \cdot PN_0 - \alpha \cdot PN_1 \quad \text{if } W = 0 \quad (1)$$

$$Y' = Y + \alpha \cdot PN_1 - \alpha \cdot PN_0 \quad \text{if } W = 1 \quad (2)$$

Step 8 : By Applying inverse DCT (IDCT) to each block after its mid-band coefficients have been modified, the CT of watermarked image is generated.

Step 9 : Finally the watermarked image can be produced by using inverse CT (ICT).

The embedding process is depicted in Fig. 3.

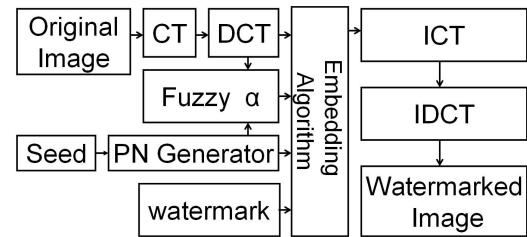


Figure 3: Watermark embedding process.

B. Watermark extraction

step 1 : Two level contourlet is applied to watermarked image and watermarked image is divided into nine sub-bands.

step 2 : Selected sub-bands are divided into $N \times N$ blocks.

step 3 : Two pseudo-random sequences PN_0 and PN_1 are regenerated by using same seeds as embedding stage.

step 4 : For each block in the selected sub-band, correlations between mid-band coefficients with PN_0 and PN_1 are calculated. If the correlation with the PN_0 is higher than the correlation with PN_1 , the watermark bit will be considered as 0, otherwise it will be considered as 1.

step 5 : After watermark extraction, similarity between the original and extracted watermarks is computed.

The extraction process is depicted in Fig. 4.

C. Fuzzy system

The used fuzzy system is a two-input and one-output system. The system tries to balance watermark power and brings us both robustness and invisibility. This fuzzy system

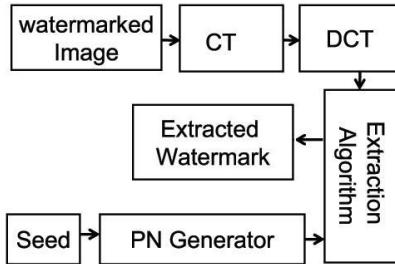


Figure 4: Watermark extraction process.

is based on a simple rule: In watermarking, large coefficients can be changed more than small ones. Two inputs of fuzzy system are named as "AVERAGE" and "DISTANCE". AVERAGE is the average of each block Y (watermark embedding:step5).

$$\text{AVERAGE}_i = \text{average}(Y_i), i = 1 \dots N_W \quad (3)$$

N_W is the number of watermark bits. To calculate the DISTANCE two correlation must be calculated. Correlation between PN_0 and Y_i which named $DIFF_0$ and correlation between PN_1 and Y_i which named $DIFF_1$. Calculation of DISTANCE is related to value of watermark bits:

$$\text{DISTANCE}_i = DIFF_0 - DIFF_1 \quad \text{if } W = 0 \quad (4)$$

$$\text{DISTANCE}_i = DIFF_1 - DIFF_0 \quad \text{if } W = 1 \quad (5)$$

$\text{DISTANCE} \in [-1.8, +1.8]$. This algorithm tries to increase the correlation of each Y_i with PN_0 or PN_1 in respect of value of W, hence fuzzy rules of system can be written as below:

- 1: If AVERAGE is very small then α is Very small.
- 2: If AVERAGE is small then α is small.
- 3: If AVERAGE is medium then α is medium.
- 4: If AVERAGE is large then α is large.
- 5: If AVERAGE is very large then α is very large.
- 6: If DISTANCE is very small then α is very small.
- 7: If DISTANCE is small then α is small.
- 8: If DISTANCE is medium then α is medium.
- 9: If DISTANCE is large then α is large.
- 10: If DISTANCE is very large then α is very large.

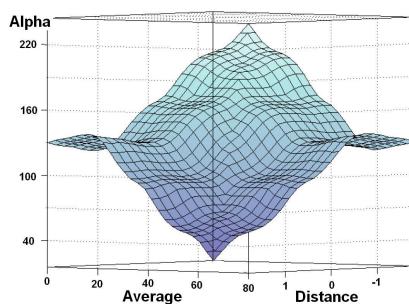


Figure 5: Fuzzy system surface.

The surface of fuzzy system is shown in Fig. 5.

D. An example

Explanation of embedding and extracting of a watermark bit into the host image is the goal of this example. As shown in Fig. 6 a grayscale, 512×512 of cat image is selected as host image. In this example, the procedure of inserting a "0" as a watermark bit into the host image will be presented.



Figure 6: Original image of Cat

1) Embedding: First Two level contourlet is applied to image and image is divided into nine sub-bands as shown in Fig. 7.

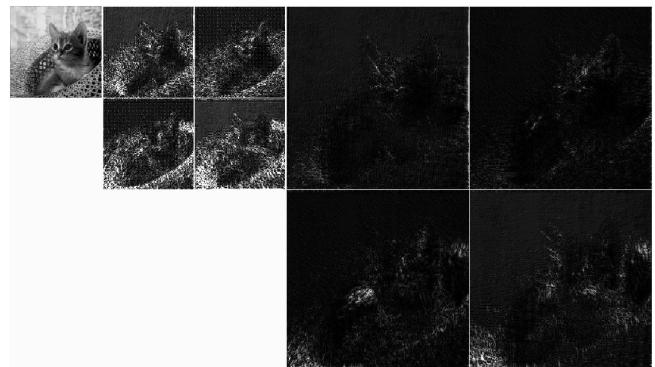


Figure 7: Contourlet representation of Cat

Then subbands 2 and 3 of second level are selected for watermark insertion (Fig. 8). Next, the selected subbands are divided into 512×512 blocks. Hence, 625 blocks per subband is obtained. In this example, the 33th bit of host image is selected for embedding.

1	2
3	4

Figure 8: Representation of subbands

DCT of selected block will be computed and its middle frequency will be selected. For further use it is named as

Mid.

$$Mid = \begin{pmatrix} 2.9323 & 0.7020 \\ 1.1322 & 1.1205 \end{pmatrix} \quad (6)$$

After that, watermark power is calculated with fuzzy system:

$$Watermarkpower = 59.2577 \quad (7)$$

Then PN_0 and PN_1 will be produced. By using equation (1), the new Mid (Mid^*) will be obtained:

$$Mid^* = \begin{pmatrix} -13.3630 & 16.9973 \\ -21.0886 & 23.3414 \end{pmatrix} \quad (8)$$

Finally, Mid will be replaced with Mid^* and all above steps will be done reversely to produce watermarked image.



Figure 9: Watermarked image of Cat

2) *Extraction*: Such as embedding section, middle frequency of 33th block ($WMid$) is produced:

$$WMid = \begin{pmatrix} -4.4099 & 14.6900 \\ -20.0518 & 14.5569 \end{pmatrix} \quad (9)$$

By calculating the correlation between this matrix and PN_0 and PN_1 , we will have :

$$\text{Correlation } (PN_0, WMid) = 0.1272 \quad (10)$$

$$\text{Correlation } (PN_1, WMid) = 0.0589 \quad (11)$$

By comparison of above results, the watermark bit will be acquired as "0".

IV. EXPERIMENTAL RESULTS

Testing this algorithm is done by using 512×512 gray scale host images: Man, Pepper, Baboon, Lena, Gold hill, Plane, Boat and Cat. Only partial results of them are shown. Watermark W is a binary pseudo-random sequence with 1225 bits length. For brevity only the pepper image is shown here. Fig. 10.

To evaluate this algorithm, we used three factors: PSNR, MSSIM and normalized Correlation Coefficient (NC). PSNR is used to evaluate the quality of watermarked image regardless of HVS, but MSSIM tries to evaluate the image quality in respect to HVS. The used PSNR formula is [10]:

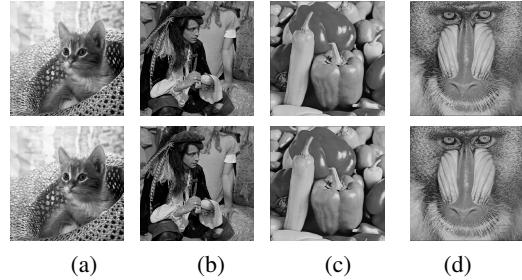


Figure 10: First row original images, second row watermarked images. (a) Cat (b) Man (c) Pepper (d) Baboon (e) Watermark

Table I: PSNR and MSSIM of watermarked images, Combination of DCT and CT can increase the quality of watermarked image.

Transform	Factor	Baboon	Cat	Man	Pepper
CT&DCT	PSNR	40.82	41.83	41.3	42.13
	MSSIM	0.990	0.985	0.984	0.988
CT	PSNR	38.48	40.13	40.45	39.85
	MSSIM	0.983	0.974	0.979	0.957

$$PSNR = 10 \log_{10} \frac{255 \times 255}{\frac{1}{L_w \cdot H_w} \sum_{x=0}^{L_w-1} \sum_{y=0}^{H_w-1} [f_{(x,y)} - g_{(x,y)}]} \quad (12)$$

where H_w and L_w are the height and width of the image, respectively. $f_{(x,y)}$ and $g_{(x,y)}$ are the values located at coordinates (x,y) of the original image, and the watermarked image, respectively.

For the sake of brevity, explanation of MSSIM is omitted. It is available in [19].

After watermark extraction, the normalized Correlation Coefficient (NC) is computed using the original watermark and the extracted watermark to measure the correctness of an extracted watermark. It is defined as [10]:

$$NC = \frac{\sum_{i=1}^{N_w} w_i \bar{w}'_i}{\sqrt{\sum_{i=1}^{N_w} w_i^2} \sqrt{\sum_{i=1}^{N_w} \bar{w}'_i^2}} \quad (13)$$

where N_w is the number of watermark bits. w and \bar{w} are the original watermark and the extracted watermark, respec-

Table II: Comparison of NC values for CT and CD-DCT.

Test	CT	CT-DCT
Gaussian filter _{5×5}	0.8858	0.9611
Median filter _{5×5}	0.7417	0.8892
Average filter _{5×5}	0.6503	0.9354
Gaussian noise (Var=20)	0.7367	0.7727
Salt and pepper (density=0.05)	0.8364	0.8353
JPEG (10%)	0.7125	0.7118
Scaling (50%)	0.9680	0.9848
Cropping (75%)	0.7066	0.7222
Histogram equalization	0.9768	0.9890
Image sharpening	0.9868	0.9912

Table III: Correlation Coefficients after attack by median , gaussian , Average filtering with various filter size ($n \times n$), and histogram equalization

Image	Median			Gaussian			Average			Hist equal
	3×3	5×5	13×13	3×3	5×5	13×13	3×3	5×5	13×13	
Cat	0.9633	0.8683	0.6703	0.9655	0.9111	0.7930	0.9650	0.8785	0.7280	0.9729
Man	0.9270	0.8009	0.6799	0.9404	0.8830	0.7630	0.9386	0.8305	0.7062	0.9599
Pepper	0.9834	0.8892	0.6888	0.9880	0.9611	0.8098	0.9879	0.9354	0.7398	0.9890
Baboon	0.9135	0.7930	0.6789	0.9289	0.8786	0.7765	0.9289	0.8363	0.7062	0.9278

Table IV: Correlation Coefficients after attack by JPEG compression with various quality and sharpening

image	JPEG								Sharpening		
	10	15	20	25	30	40	50	65	75	80	
Cat	0.7797	0.8317	0.8654	0.8940	0.9243	0.9576	0.9689	0.9756	0.9790	0.9810	0.9857
Man	0.7591	0.8029	0.8452	0.8884	0.9066	0.9248	0.9311	0.9386	0.9445	0.9453	0.9513
Pepper	0.7118	0.7535	0.8310	0.8873	0.9165	0.9723	0.9846	0.9885	0.9881	0.9910	0.9912
Baboon	0.7928	0.8469	0.8943	0.9112	0.9135	0.9279	0.9348	0.9406	0.950	0.9505	0.9366

Table V: Correlation Coefficients after attack by gaussian noise which its variance varies from 1 to 10, salt & pepper noise which varies from 2% to 20% , cropping up to 75% of image and scaling 50% and 75%

image	Salt and Pepper			Gaussian noise			Cropping (%)			Scaling (%)	
	0.02	0.05	0.2	1	10	20	20	50	75	50	75
Cat	0.9042	0.8424	0.7314	0.9366	0.7757	0.7218	0.9383	0.8346	0.7140	0.9582	0.9769
Man	0.8837	0.8244	0.7337	0.9031	0.7730	0.7277	0.91148	0.8248	0.6988	0.9294	0.9450
Pepper	0.9090	0.8353	0.7105	0.9320	0.7727	0.7192	0.9500	0.8452	0.7222	0.9848	0.9890
Baboon	0.8894	0.8218	0.7430	0.9044	0.7956	0.7334	0.9009	0.8184	0.7017	0.9223	0.9406

Table VI: Comparison between proposed method and some previous works.

Test	Shao zho[10]	Zhao Xu [11]	XIE Jing[12]	Proposed method
Gaussian filter 3x3	0.9321	0.7658	0.8956	0.9861
Median filter 3x3	NA	NA	0.9689	0.9868
Gaussian noise (Var=0.001)	0.9683	0.8430	0.4505	0.9829
Salt and pepper (density=0.001)	0.9976	0.98	0.9472	0.9853
Speckle noise (density=0.001)	NA	NA	0.9472	0.9865
Scaling (40%)	0.9866	0.8394	NA	0.9041
Cropping (50%)	0.7244	0.7240	NA	0.7835
JPEG (80%)	1.000	0.9870	NA	0.9850
JPEG (50%)	1.000	0.8670	0.8956	0.9838
Histogram equalization	NA	NA	NA	0.9812
Image sharpening	NA	NA	NA	0.9861

tively. The watermarked images and extracted watermark have been shown in Fig. 11.

In experiments both geometric and non-geometric attacks are considered. Non-geometric attacks includes JPEG compression, histogram equalization, sharpening and gaussian, median and average filtering and Gaussian noise, salt and pepper noise. For geometric attacks, scaling and cropping are used. The results are shown in Tables III, V, and IV. A comparison between proposed method and some previous works is shown in Table VI, in this comparison "Lena" image is used as host [10], [11], [12].

A group of 10,000 different watermarks including the genuine (embedded) one is used for evaluating the robustness of watermark detection algorithm against various attacks. As mentioned before each watermark is a 1225-bit

binary pseudo-random sequence. The experimental results show that for genuine watermark the detector algorithm has highest response (the genuine watermark is 5000th watermark).

Detection responses of algorithm against various attacks and relevant images are shown in Fig. 11 and Fig. 12, respectively, and prove that proposed method is robust against both geometric and non-geometric attacks.

V. CONCLUSION

A blind watermarking scheme was proposed in this paper, which uses Contourlet and Discrete Cosine Transform to increase robustness and invisibility. By considering the fact that human eyes are less sensitive to noise in oblique angels, Contourlet was used because of its directional property. On

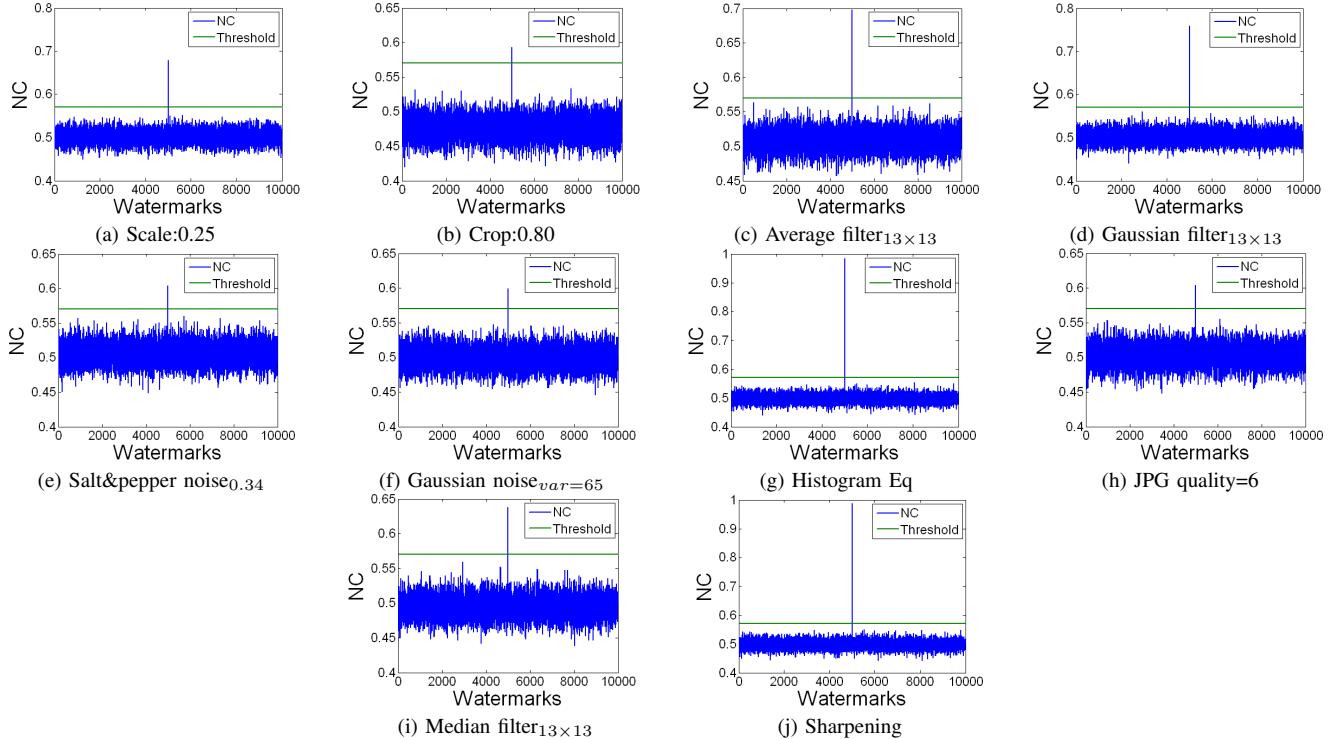


Figure 11: Detector response against various attacks

the other hand to improve the throughput of watermarking algorithm, the power of watermark is determined with a two-input fuzzy system. Experimental results demonstrate that the proposed scheme is robust against common non-geometric and geometric attacks.

REFERENCES

- [1] S. Haohao, S. Yu, X. Yang, L. Song, and C. Wang, "Contourlet-based image adaptive watermarking," *Signal Processing: Image Communication*, vol. 23, no. 23, pp. 167–178, 2008.
- [2] M. Prasad. R and Sh. Koliwad, "A comprehensive survey of contemporary researches in watermarking for copyright protection of digital images," *IJCSNS International Journal of Computer Science and Network Security*, vol. 9, no. 4, April 2009.
- [3] G. Voyatzis and I. Pitas, "The use of watermarks in the protection of digital multimedia products," in *Proceedings of IEEE*, 1999, vol. 87.
- [4] R. Wolfgang, C. Podilchuk, and E. Delp, "Perceptual watermarks for digital images and video," in *Proceedings of IEEE*, July 1999, vol. 87, pp. 1108–1126.
- [5] M. Barni, F. Bartolini, and A. Piva, "Improved wavelet-based watermarking through pixel-wise masking," *IEEE Transactions on Image Processing*, vol. 10, no. 5, pp. 783–791, 2001.
- [6] M.J. Tsai, Ch.T. Lin, and J. Liu, "A wavelet-based watermarking scheme using double wavelet tree energy modulation," in *ICIP*, 2008, pp. 417–420.
- [7] X.B. Wen, H. Zhang, X.Q. Xu, and J.J. Quan, "A new watermarking approach based on probabilistic neural network in wavelet domain," *Soft Comput.*, vol. 13, no. 4, pp. 355–360, 2009.
- [8] H. Li, W. Song, and Sh. Wang, "A novel blind watermarking algorithm in contourlet domain," in *18th International Conference on Pattern Recognition*, 2006, vol. 3, pp. 639–642.
- [9] S. Zaboli and M.S. Moin, "Cew: A non-blind adaptive image watermarking approach based on entropy in contourlet domain," in *IEEE International Symposium on Industrial Electronics*, June 2007, pp. 1687–1692.
- [10] S. M. Zhu and J. M. Liu, "A novel blind watermarking scheme in contourlet domain based on singular value decomposition," *International Workshop on Knowledge Discovery and Data Mining*, vol. 0, pp. 672–675, 2009.
- [11] Z. Xu, K. Wang, and X. h .Q, "A novel watermarking scheme in contourlet domain based on independent component analysis," *Intelligent Information Hiding and Multimedia Signal Processing, International Conference on*, vol. 0, pp. 59–62, 2006.
- [12] J. Xie and Y. Wu, "Fingerprint image watermarking algorithm using the quantization of parity based on contourlet transform," *Computer Applications*, vol. 6, pp. 1365–1367, 2007.
- [13] M.N. Do and M. Vetterli, "The contourlet transform:an efficient directional multiresolution image representation," in *IEEE Trans. on Image Processing*, December 2005, vol. 14, pp. 2091–2106.
- [14] E.J. Candes and D. Donoho, "New tight frames of curvelets and optimal representations of objects with smooth singularities," in *technical report*, 2002.
- [15] E.L. Pennec and S. Mallat, "Sparse geometric image representation with bandelets," in *IEEE Trans. on Image Processing*, April 2005, vol. 14, pp. 423–438.
- [16] M.N. Do and M. Vetterli, "Framing pyramids," in *IEEE Trans. on Signal Processing*, September 2003, vol. 51, pp. 2329–2342.



Figure 12: Some used images for evaluating the robustness of algorithm against various attacks

- [17] R.H. Bamberger and M.J.T. Smith, “A filter bank for the directional decomposition of images: Theory and design,” in *IEEE Trans. on Signal Processing*, April 1992, vol. 40, pp. 882–893.
- [18] Glenn and E. William, “Digital image compression based on visual perception,” pp. 63–71, 1993.
- [19] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error measurement to structural similarity,” *IEEE Trans. on Image Processing*, vol. 13, pp. 600–612, 2004.

A Secure Database System using Homomorphic Encryption Schemes

¹Youssef Gahi, ²Mouhcine Guennoun, ²Khalil El-Khatib

¹Ecole Mohammadia d'Ingénieurs B.P 765 Avenue Ibn Sina, Agdal,
Rabat, Morocco

²University of Ontario Institute of Technology, 2000 Simcoe Street North,
Oshawa, Ontario, Canada. L1H 7K4

youssef.gahi@gmail.com, mouhcine.guennoun@uoit.ca, khalil.el-khatib@uoit.ca

Abstract—Cloud computing emerges as an attractive solution that can be delegated to store and process confidential data. However, several security risks are encountered with such a system as the securely encrypted data should be decrypted before processing them. Therefore, the decrypted data is susceptible to reading and alterations. As a result, processing *encrypted* data has been a research subject since the publication of the RSA encryption scheme in 1978. In this paper we present a relational database system based on homomorphic encryption schemes to preserve the integrity and confidentiality of the data. Our system executes SQL queries over encrypted data. We tested our system with a recently developed homomorphic scheme that enables the execution of arithmetic operations on ciphertexts. We show that the proposed system performs accurate SQL operations, yet its performance discourages a practical implementation of this system.

Keywords-Private Information Retrieval; Secure Database; Homomorphic Encryption Schemes; Privacy.

I. INTRODUCTION

Cloud computing is an attractive solution that can provide low cost storage and processing capabilities for government agencies, hospitals, and small and medium enterprises. It has the advantage of reducing the IT costs and providing more services for the requesting parties through making specialized software and computing resources available. However, there are major concerns that should be considered by any organization migrating to cloud computing. The confidentiality of information as well as the liability for incidents affecting the infrastructure arise as two important examples in this context. Indeed, cloud computing poses several data protection risks for the cloud's clients and providers. For example, the cloud's client may not be aware of the practices according to which the cloud's provider processes the stored data. Therefore, the cloud's client cannot guarantee that the data are processed (for example, altered or deleted) in a legal and accepted manner.

All of the above mentioned issues can be resolved if the data in the cloud are stored and processed in encrypted form. The latter is possible if the encryption scheme can support *addition* and *multiplication* of the encrypted data. Many encryption schemes support one of these operations, like the encryption schemes in [1-4]. A cryptosystem which supports both addition and multiplication (referred to as the homomorphic encryption scheme) can be effective data protection, and enables the construction of programs that receive encrypted input and produce encrypted output. Since

such programs do not decrypt the input, they can be run by an un-trusted party without revealing their data and internal states. Such programs will have great practical implications in the outsourcing of private computations, especially in the context of cloud computing.

Homomorphic cryptosystems have received valuable attention in the literature, see [5][6][7]. In theory, the data can be encrypted by the client, and then sent to the cloud's provider for storage or processing. Only the client holds the decryption keys necessary to read the data. Despite the fact that this type of processing may increase the amount of computing time, the benefits associated with it are worth the processing overhead. Indeed, this model of computing can preserve the confidentiality and integrity of the data while delegating the storage and processing to an un-trusted third party.

In this paper, we present a novel technique to execute SQL statements over encrypted data. We develop a secure database system that processes these queries. The parameters of SQL queries are encrypted by the client and sent to the server for processing. The latter performs the requested operation over an encrypted database and returns an encrypted result to the client. The advantage of this system is that the database server knows neither the content nor the position of the records affected by the query.

The remainder of this paper is organized as follows. In Section II, we review the literature for the work related to private information retrieval (PIR) approaches. Section III provides a formal description of our secured SQL statements approach. Section IV presents a homomorphic cryptosystem that we use to build a prototype system. Section V presents an implementation of a secure relational database system. In Section VI we provide performance analysis of the proposed secure database system. Finally, Section VII concludes our work and provides future research directions.

II. PRIVATE INFORMATION RETRIEVAL

Chow et al. [8] discussed the importance of cloud computing, and how this technology can be enticing due to its flexibility and cost-efficiency. The authors pointed out that the adoption of such technology is still below ambition. Some users are still concerned about the security of these clouds. Even those who started using the technology, they only utilize it with their less sensitive data. The limited usage of cloud computing is mainly due to the lack of control over the communicated data. The authors highlight that people require explicit guarantees that their data will be protected under well-defined policies and mechanisms. However, no

technical security solutions were proposed to back-up their *information centric* model where data can self defend itself in a hostile or an un-trusted environment.

The private information retrieval (PIR) approach, introduced by *Chor et al.* in [9], achieves the retrieval of an i^{th} bit in a block without revealing information about the bit retrieved or about the request for the bit itself. This approach has been widely used as a basis for several tools, and has supported various distributed applications. However, the approach requires more improvements and the work with it is still in progress, both at the security of the communication channel level and the hidden client identity level.

Raykova et al. [10] extended the PIR approach by proposing a secure anonymous search system. The system employs keyword search such that only authorized clients have access to their blocks. This system is capable of mapping the database content to the appropriate client, thus guaranteeing the privacy of the data and the query. The ultimate target of *Raykova's* system is to ignore the identity of the client while protecting the database from malicious queriers.

Shang et al. [11] tackled the problem of protecting the database itself. The problem is studied through monitoring the amount of data disclosed by a PIR protocol during a single run. The information attained from the monitoring process is used to understand how a malicious querier can conduct attacks to retrieve excessive amount of data from the server.

PIR has also been used to develop authentication systems. *Nakamura et al.* [12] constructed a system with three components, a querier that initiates requests, an authentication-server that processes these requests, and a database that returns the appropriate data in response to the request. This system ensures the security of data and the anonymous communication between the querier and the database. *Yinan and Cao* [13] used the PIR approach to propose a system that controls the access to the database. According to this system, the privacy of data is enforced by enabling each authorizer to give or deny access to his/her own data with a hierarchical authorization access right scheme.

Among the most important criteria in PIR protocol are the communication cost and the amount of data sent back to the querier. The trivial solution of the PIR protocol is to send back the entire database to the client. However, this solution is expensive, even for a simple request that results in retrieving two matching records. Other approaches proposed to retrieve only the requested data, by using replicated databases that are stored at multiple servers. In this case, the request is forwarded to all servers. With this approach, although we deal with multiple replicated databases, the privacy is better protected. However, this approach is still complicated and may result in extended processing and communication times. *Gentry et al.* [14] proposed a scheme to retrieve a bit or a block from a database with a constant communication rate. *Melchor et al.* [15] proposed a scheme that reaches the available data with a reasonable communication cost while achieving lower computational cost compared to other PIR protocols.

III. SECURED SQL OPERATIONS

In this section, we develop a secure database system that processes SQL queries over encrypted data. As shown in Figure 1, parameters of the queries are encrypted by the client and sent to the server for processing. The latter performs the requested operation and returns encrypted results to the client.

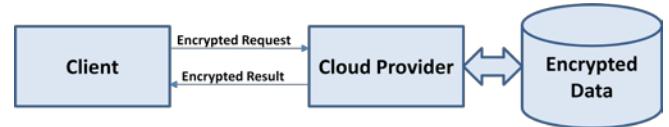


Figure 1. Secure Data Retrieval

We describe below the circuit of a simple SQL SELECT query:

`SELECT * from T where c=v`

where the value v is in encrypted form. The trivial solution to securely perform this statement is to send back to the client the entire database, but this solution suffers from complexity and scalability issues. Instead, we propose a methodology to implement the SELECT circuit at the server side, while preserving the confidentiality and the privacy of the request.

The processing of the SELECT query is divided into three sub-circuits. Firstly, we calculate the following index for each record R in the table T :

$$\forall R \in T I_R = \prod_{i=0}^{size-1} (1 \oplus c_i \oplus v_i)$$

where $size$ is the number of bits in column c ; c_i and v_i are the i^{th} bits of column c and search criteria v , respectively. I_R is a one bit value that is equal to 1 if v matches the value of column c , 0 otherwise.

Next, we identify the n^{th} record that matches the selection criteria. For that purpose, we consider $\eta = \varepsilon_{pk}(n)$ to be the encryption of n under public key pk .

For each record R we calculate the following sum:

$$\forall R \in T: S_R = \sum_{i \leq R} I_R$$

We calculate a second index I'_R :

$$\forall R \in T I'_R = I_R \times \prod_{i=0}^{size-1} (1 \oplus \eta_i \oplus S_{R,i})$$

I'_R is equal to 1 if the record R is the n^{th} record that matches the selection criteria, 0 otherwise.

Then, we multiply every bit of each record R in table T by the corresponding value I'_R .

$$\forall R \in T: R' = I'_R \times R$$

This latter operation forms a table T' that is related to the original table T as follows:

$$\begin{cases} R' = R \text{ if } R \text{ is the } n^{\text{th}} \text{ record matching the criteria} \\ R' = 0 \text{ otherwise} \end{cases}$$

Finally, by adding all records of table T' , we retrieve the n^{th} record R_s that matches the selection criteria:

$$R_s = \sum_{R' \in T'} R'$$

If no record matches the selection criteria, a record containing zeros will be returned to the requester.

It is worth noting that all calculations are performed over encrypted data. The server does a blind processing to retrieve the n^{th} record that matches the selection criteria. It neither has access to the content of the retrieved record nor to its position within the table.

With slight modifications to the select circuit, most of SQL operations can be supported by our proposed secure database system. For example, to implement the UPDATE operation, one can simply implement the following circuit:

$$\forall R \in T: R' = \overline{I_R} * R + I_R * U$$

where the record U is the new value to update the record R matching the criteria of the query.

Similarly, to delete a record from table T , one can replace its content by zero. The DELETE operation can be implemented by the following circuit:

$$\forall R \in T: R' = \overline{I_R} * R$$

IV. HOMOMORPHIC ENCRYPTION SCHEME

In [6], Gentry proposed a fully homomorphic encryption scheme that enables to perform an arbitrary number of arithmetic operations (i.e. addition and multiplication) on encrypted data. The components of the encryption scheme are described below.

Security Parameters: $N = \lambda$, $P = \lambda^2$, and $Q = \lambda^5$.

A. Key Generation

The private key s_k is a random P-bit odd number. The public key consists of a list of integers that are the “encryptions of zero” using the encryption scheme with the secret key s_k as a public key.

Generate a set $\vec{y} = \{y_1, \dots, y_\beta\}$ of rational numbers in $[0, 2]$ such that there is a sparse subset $S \subseteq \{1, \dots, \beta\}$ of size α with $\sum_{i \in S} y_i \approx 1/p \bmod 2$.

Set sk^* to be the sparse subset S , encoded as a vector $s \in \{0, 1\}^\beta$ with hamming weight α .

Set $pk^* \leftarrow (pk, \vec{y})$ to be the public key.

B. Encryption (pk^*, m)

Set m' to be a random N-bit number such that m and m' have the same parity:

$$m' = m \% 2$$

Then compute c as:

$$c \leftarrow m' + pq$$

where q is a random Q-bit number. Then the ciphertext c is post-processed to produce a vector $\vec{z} = \{z_1, \dots, z_\beta\}$, defined by:

$$z_i \leftarrow c \cdot y_i \bmod 2$$

The output ciphertext c^* consists of c and $\vec{z} = \{z_1, \dots, z_\beta\}$.

C. Decryption (sk^*, c^*)

$$m \leftarrow LSB(c) \oplus LSB(\sum_i s_i \cdot z_i)$$

D. Arithmetic Operations

Addition and multiplication can be performed on clear text by simply adding and multiplying the ciphertexts, respectively.

$$\varepsilon_{pk}(m_1 * m_2) = \varepsilon_{pk}(m_1) * \varepsilon_{pk}(m_2)$$

$$\varepsilon_{pk}(m_1 + m_2) = \varepsilon_{pk}(m_1) + \varepsilon_{pk}(m_2)$$

The output ciphertext c^* consists of c together with the result of post-processing the resulting ciphertext with \vec{y} .

E. Bootstrapping the Encryption Scheme

The scheme described above is referred to as a *somewhat* homomorphic scheme because it works only if the value $c \% p$ (noise of the encryption) is smaller than $p/2$. After a finite number of arithmetic operations, the noise exceeds the $p/2$ threshold and the decryption scheme does not work anymore.

Gentry developed a novel method to remove the noise in the ciphertext [7]. He proposed to recrypt the ciphertext c to remove the noise. Since the scheme is homomorphic, one can encrypt the ciphertext c into a new ciphertext \bar{c} (the plaintext is encrypted twice), and by using the homomorphic properties of the scheme, one can decrypt the inner layer of encryption to obtain a ciphertext c_2 with a lower value of noise.

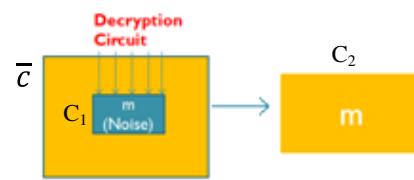


Figure 2. Removing noise from original ciphertext (bootstrapping)

As illustrated in Figure 2, a bit m is encrypted with public key pk to produce the ciphertext c_1 . After a finite number of arithmetic operations, the noise associated with the ciphertext c_1 reaches a level that does not permit any additional arithmetic operation. To remove the noise, the bootstrapping technique consists of recrypting the bit m . Every bit of ciphertext c_1 is encrypted with the public key pk . The output is ciphertext \bar{c} that doubly encrypts bit m . The decryption circuit is applied to remove the inner layer of encryption. This latter operation requires the knowledge of the private key sk . Therefore, the private key is encrypted with public key pk ; and then shared with the server. Since the encryption scheme is homomorphic, the decryption can be performed on the doubly encrypted ciphertext to remove the inner layer. The recryption produces a new ciphertext c_2 with a value of noise that has an upper bound according to the proof in [6].

By employing the bootstrapping technique, performing an arbitrary number of arithmetic operations on ciphertexts becomes possible.

V. IMPLEMENTATION

In our implementation, we aim at proving that it is possible to perform SQL queries over an encrypted database. For example, the user can specify a search criterion through a database. Then, the client software encrypts the parameters of the query, corresponding to the search criterion, and sends it to the appropriate server. The server retrieves the requested record (blind processing) from the database and returns it to the client. The client software decrypts the record and displays it to the user.

We built a simple medical application containing 10 patients' records. In Figure 3, we show the result of the SELECT query. This is how the result appears in a screenshot of the client side of our built application.

The application supports the following SQL operations:

- SELECT with wildcard characters (*, ?) and relational operators (< >).
- UPDATE with wildcard characters (*, ?) and relational operators (< >).
- DELETE with wildcard characters (*, ?) and relational operators (< >).
- Statistical operations like COUNT and AVG.

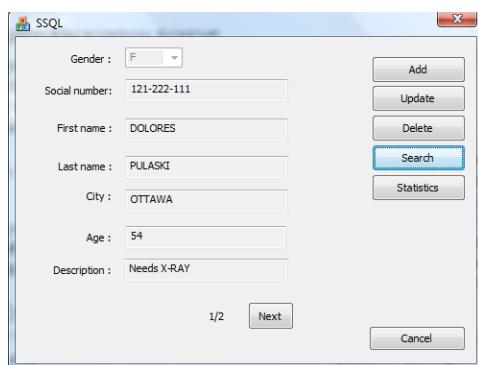


Figure 3. Client side of the application

It is worth mentioning that the implementation of the medical application was built using a simplified and non-secure version of the somewhat homomorphic scheme. This is due to performance issues as it is impractical to perform our tests using the fully homomorphic cryptosystem. We chose the security parameters in such a way to support all the SQL operations with no need to employ the bootstrapping technique. We discuss the performance of our system in the next section.

VI. PERFORMANCE ANALYSIS

Table 1 lists the number of arithmetic operations required to execute some basic SQL statements over an encrypted database of 10 records. From this table we can see that processing data in encrypted form creates a substantial computation overhead.

TABLE I. NUMBER OF ARITHMETIC OPERATIONS

	Add. & Mult.	Add.	Mult.
SELECT	619839	309892	309947
UPDATE	67595	25355	42240
DELETE	28171	5643	22528

To understand the processing time required to process a SQL statement, we measured the time required to perform the product of two n-bits numbers in encrypted form using the fully homomorphic cryptosystem presented in [6]. Towards that end, we used a computer machine with 1.7 GHz processor and 3GB of RAM memory. Figure 3 shows the amount of time, in seconds, required to compute the multiplication circuit.

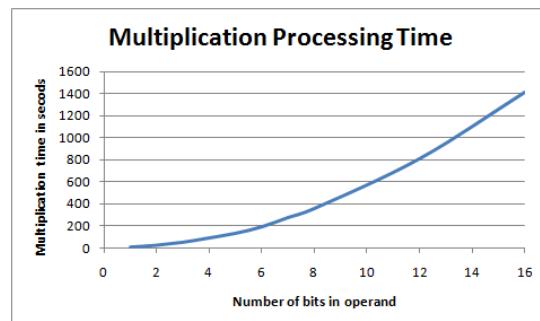


Figure 4. Processing time required to perform the product of two n-bits integers

As we can see in Figure 4, it takes 23 minutes to compute the product of two 16-bit integers. This latency is due to the bootstrapping technique or more precisely to the recrypt function. Indeed, according to our measurements, it takes 1 second to recrypt a ciphertext. Therefore, there is a need of at least 7 days (i.e. $619839 * 1$ second) to retrieve a row from a 10-record database.

The implementation of the system proves that the execution of SQL statements over encrypted data is feasible.

However, the time required to execute these statements is very high and therefore is not suitable for real-time transactions that involve a large database (i.e. several terabytes database). This drawback is mainly due to the homomorphic encryption scheme. In fact, there might be more efficient techniques to optimize the implementation, that is, one could perform recryption only when it is necessary, since the noise value can be bounded; however, we do believe that a more practical homomorphic cryptosystem is yet to be developed.

VII. CONCLUSION & FUTURE DIRECTIONS

The concept of processing encrypted data is promising to revolutionize traditional computing. Indeed, this concept has many direct applications in cloud computing environments, banking, electronic voting and many other applications.

In this paper we developed the first secure database system based on a fully homomorphic encryption scheme. We presented the circuits to implement SQL statements over encrypted data. We built a prototype of a database system where data is stored and processed in encrypted form. The database server can execute most of the SQL statements in a blind fashion, that is, it returns the results without any knowledge of the content or the position of the records extracted/affected. We conducted performance analysis to measure the time needed to execute a simple query on the database. We found that the current technology is not sufficiently mature yet as it is time-consuming. Indeed, the encryption schemes proposed by *Gentry et al.* in [5][6][7] are very impractical. According to our measurements, the time needed to perform simple calculations is substantial. We believe that there still is a great opportunity for researchers to develop more efficient homomorphic encryption schemes.

As future work, we are planning to work on the optimization of the efficiency of the system. Processing can be parallelized in order to take advantage of multiple processors executing the encrypted requests. We will also investigate how to reduce the number of recryptions needed. Indeed, since the noise value can be bounded, decryption should be necessary only when the ciphertext cannot support an additional arithmetic operation. We will be planning to develop a new scheme to encrypt the SQL circuits. In the current system, the server does know the operation that was performed (SELECT, UPDATE, etc.). If we can encrypt the SQL circuits, the system will preserve the confidentiality of

the data and operations performed on these data. We believe that this new system can be the foundation of a highly secure cloud computing environment.

REFERENCES

- [1] R. Rivest, A. Shamir, and L. Adleman, A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, Communications of the ACM 21 (2): pp. 120–126, 1978.
- [2] T. ElGamal, A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms, IEEE Transactions on Information Theory, pp. 469–472, 1985.
- [3] S. Goldwasser and S. Micali, Probabilistic Encryption. Journal of Computer and System Sciences, 28(2): pp. 270-299, April 1984.
- [4] P. Paillier, Public-Key Cryptosystems Based on Composite Degree Residuosity Classes, Advances in Cryptology — EUROCRYPT '99 In Advances in Cryptology — EUROCRYPT '99 , Vol. 1592 (1999), pp. 223-238, 1999.
- [5] M. V. Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, Fully Homomorphic Encryption over the Integers. EUROCRYPT 2010: pp. 24-43, June 2010.
- [6] C. Gentry, Computing arbitrary functions of encrypted data, Commun. ACM, Vol. 53, No. 3., pp. 97-105, March 2010.
- [7] C. Gentry, A fully homomorphic encryption scheme. PhD thesis, Stanford University, 2009.
- [8] R. Chow, P. Golle, M. Jakobsson, R. Masuoka, and J. Molina, Controlling Data in the Cloud : Outsourcing Computation without Outsourcing Control. CCSW'09, pp. 85-90, Chicago, Illinois, USA, November 13, 2009.
- [9] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, Private Information Retrieval, Journal of the ACM, 45(6): pp. 965-982, 1998.
- [10] M. Raykova, B. Vo, and S. Bellovin, Secure Anonymous Database Search, CCSW'09, pp. 115-126, Chicago, Illinois, USA, November 13, 2009.
- [11] N. Shang, G. Ghinita, Y. Zhou, and E. Bertino, Controlling Data Disclosure in Computational PIR Protocols. ASIACCS'10, pp. 310-313, Beijing, China, April 13–16, 2010.
- [12] T. Nakamura, S. Inenaga, D. Ikeda, K. Baba, H. Yasuura, Anonymous Authentication Systems Based on Private Information Retrieval. Networked Digital Technologies. NDT '09, pp.53-58, 28-31 July 2009.
- [13] S. Yinan and Z. Cao, Extended Attribute Based Encryption for Private Information Retrieval. Mobile Adhoc and Sensor Systems, 2009. MASS '09, pp. 702-707, 12-15 Oct. 2009.
- [14] C. Gentry and Z. Ramzan, Single-Database Private Information Retrieval with Constant Communication Rate. ICALP 2005, LNCS 3580, pp. 803–815, 2005.
- [15] C. A. Melchor and P. Gaborit, A Fast Private Information Retrieval Protocol. ISIT 2008, pp. 1848-1852, Toronto, Canada, July 6 - 11, 2008.

SQRM: An Effective Solution to Suspicious Users in Database

Dai Hua

College of Information Science & Technology, Nanjing
 University of Aeronautics & Astronautics
 Nanjing, China
 dai_hua@nuaa.edu.cn

Zheng Guineng

College of Information Science & Technology, Nanjing
 University of Aeronautics & Astronautics
 Nanjing, China
 redzgn@nuaa.edu.cn

Abstract—Since traditional database mechanisms such as identity authentication and access control, can be fooled by authorized but malicious users, to solving the problems, three key techniques namely intrusion detection, damage quarantine and recovery are studied for decades to implement survival database systems. However, these techniques are all built on identification of malicious behaviors, which is much more complex, sluggish and inefficient than the identification of suspicious behaviors because the former need more evidence than the later. This paper proposes an effective security mechanism by focusing suspicious users, namely suspect quarantine and recovery method denoted as SQRM, to increase the attack resistance of databases. It isolates invalid data transparently from trustworthy users to prevent further damage by suspicious users suspected to be malicious, while still maintaining continued availability for their data access operations to minimize loss of productive work in the case of incidents that they are indeed innocent. And when they are proved innocent or malicious, all invalid data caused by them will be concurrently recovered. Using SQRM is sufficiently effective to improve the survivability for database.

Keywords-database security; survival database; suspicious user quarantine; invalid data recovery

I. INTRODUCTION

Database security, an issue focuses on data confidentiality, integrity and availability [1] has drawn a considerable amount of interest since database was used in data-intensive and security-sensitive applications, such as credit card billing, banking, air traffic control and online stock trading. Traditional database security technologies, such as identity authentication, access controls and encryption concentrate on database confidentiality, which is often powerless for malicious attacks including authorized abusing, hackings, and so on. So many attacks succeeded, which had fooled traditional database protection mechanisms, because in reality not all attacks can be averted at their beginning. Consequently, survival database systems (or attack resistant, or intrusion tolerant, or self healing database systems) [2-5] are of significant concern, which can survive malicious attacks, and provide continuous but

Qin Xiaolin

College of Information Science & Technology, Nanjing
 University of Aeronautics & Astronautics
 Nanjing, China
 qinxcs@nuaa.edu.cn

Li Ziyue

College of Information Science & Technology, Nanjing
 University of Aeronautics & Astronautics
 Nanjing, China
 yenuo_1108@msn.com

maybe degraded service when the damage is being recovered.

To implement survival database systems, three key technologies namely intrusion detection (ID) [6-9], damage quarantine (DQ) [10-14] and damage recovery (DR) [15-19] have been studied for decades. ID detects malicious attacks including malicious users' transactions and operations. DQ isolated all invalid data result from corruption of malicious attacks detected by ID, and ensure invalid data not be accessed by trustworthy users, otherwise it might cause damage spreading [20] (if data x is invalid, operation $y=x+100$ will have damage spread to y). DR repairs all invalid data and improves availability of database. Apparently, ID, DQ and DR are built on the identification of malicious behaviors. Actually, the identification of suspicious behaviors could be more efficient, easier and earlier than identification of malicious behaviors in practical applications, because the latter needs more evidence to investigate. Obviously trustworthy data would be in danger as long as the suspicious behaviors exist because they could be indeed malicious. Therefore if we can control suspicious behaviors immediately after it has been detected, the indeed malicious attacks will be prevented earlier; the scare of damage will be decreased and the recovery of database will be easier and more efficient.

Here, we focus on suspicious users, whose behaviors are suspicious, but still need further investigation and more evidence to finally confirm their uncertain identities innocent or malicious. For example, when an accountant logs on banking system at 2:00 am as user "Jack" who usually works in the daytime, this abnormal logon will make Jack suspicious. The real identity of this Jack is uncertain. Perhaps Jack himself is working overtime involving an urgent task, or this Jack is a malicious hacker who cheated jack's identity. More evidence is needed to make the right judgment. What could we do if we encounter this suspicious Jack? The naive rejection would cause loss of his constructive work if he is indeed Jack himself. On the other hand, the simple permission may cause further damage if he is a hacker. As a result, to handle the above dilemma, necessary measures should be taken toward suspicious users.

In order to solve problems of suspicious users, Suspect Quarantine and Recovery Method (SQRM) is presented by us in this paper. SQRM has two phases of work: Suspicious User Quarantine Phase (SUQ-Phase) and Invalid Data Recovery Phase (IDR-Phase). As shown in Figure 1, user s was trustworthy before it was detected suspicious at time t_1 , proved innocent or malicious at time t_2 , and invalid data recovery was accomplished at time t_3 . SUQ-Phase starts from t_1 to t_2 , while IDR-Phase originates from t_2 to t_3 . The key points of SQRM are as follows:

- In SUQ-Phase, s will be quarantined immediately once s is detected suspicious, but instead of being stopped arbitrarily, s will be able to continue its work. An extra value of data is provided to s for its data accessing. Meanwhile, the invalid data caused by s will be access denied by trustworthy users to prevent damage spreading since it is recovered.
- In IDR-Phase, when s is proved innocent or malicious, all the extra value of data caused by s will be identified. If s is proved malicious, the extra value of data caused by s will be incorrect and discarded directly, but if s is proved innocent, it will be identified as correct and written back into the database. All the invalid data caused by s will be recovered at last.

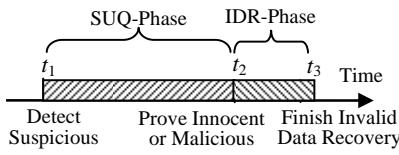


Figure 1. SQRM workflow

There is an evaluation criterion of judging the strategy of handling suspicious users: No Leakage of Invalid Data (NLID). NLID requires that invalid data should be isolated from trustworthy users, which means that they would not access any invalid data, so the damage spreading will be prevented. Meanwhile all invalid data will be recovered trustworthy, and the integrity and correctness of database will be assured.

To satisfy the NLID criterion and make sure suspicious users working under quarantine, we present a data model of SQRM firstly, which characterizes the value types of data items maintained by trustworthy and suspicious users. Then we provide the user operation isolation algorithm and on-the-fly invalid data recovery algorithm based on the data model, the former algorithm will not only isolate all invalid data from trustworthy users to prevent damage spreading, but also provide extra value of data items to suspicious users to continue their work, while the later will recover all the invalid data in IDR-Phase of suspicious users.

The rest of the paper is organized as follows. Section 2 discusses the related works. In Section 3, we give the theoretical model and algorithms of SQRM. Finally, Section 4 summarizes what we have done and future work of this paper.

II. RELATED WORKS

Since current research mostly relies on ID, DQ and DR method to solve malicious attacks to implement survival database systems. Only a few studies of suspicious users have been proposed. In current research of suspicious uses, Liu et al. proposed a data attack isolation system (DAIS) using data versions [21-23]. The main point of DAIS includes two steps as following: In step 1, once a user s is detected suspicious, it will be isolated from other users according to isolation protocol, and suspicious version data will be created and maintained by s when s performs updates in database. Meanwhile trustworthy users can not access any suspicious version data. In step 2, when s is proved malicious, all the suspicious version data maintained by s will be discarded. And when s is proved innocent, to resolve the conflicts between trustworthy and innocent transactions statically or dynamically, precedence graph of transactions will be created. Because the acyclic precedence graph means no conflict of transactions and consistence of database, if cycles appear in precedence graph, related committed transactions (trustworthy or innocent) incurring cycles will be backed out to break cycles thus guaranteeing precedence graph acyclic (Back out a transaction means to restore every data item updated by it to the latest value before updates). After precedence graph is established, the suspicious version data (which is indeed trustworthy) maintained by the innocent user s will be adopted to replace the corresponding trustworthy version data (which are indeed invalid). Till now the processing of suspicious user s is accomplished.

However, DAIS still has shortcomings in damage spreading. We illustrate it by giving an example as follows:

Example 1: Suppose user s is detected suspicious at time t_1 , proved innocent at time t_2 , and user u is always trustworthy. During time interval $[t_1, t_2]$, s executes transactions T_{s1} and T_{s2} while u executes T_{u1} , T_{u2} and T_{u3} . Details of these transactions are shown in Figure 2. We denote trustworthy version data as $x[T]$, and suspicious version data as $x[s]$ maintained by s .

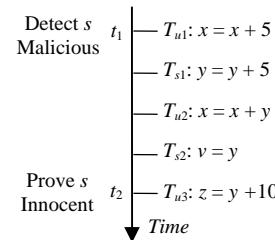


Figure 2. Operations and History of Transactions

In $[t_1, t_2]$, s is suspicious. According to DAIS, when 5 transactions are finished, $x[T]$ and $z[T]$ will be updated while $y[s]$ and $v[s]$ are created ($x[T]=21$, $z[T]=18$, $y[s]=13$ and $v[s]=13$ as shown in TABLE I). When s is proved innocent at t_2 , an acyclic precedence graph G of committed transactions will be created as shown in Figure 3. Obviously there is no conflict of transactions. Since the proved

innocence of s indicates that T_{s1} and T_{s2} are trustworthy, $y[s]$ and $v[s]$ are actually trustworthy too, they will be adopted to replace $y[T]$ and $v[T]$ ($y[s]=13$ replace $y[T]=8$ and $v[s]=13$ replace $v[T]=8$, then remove $y[s]$ and $v[s]$ as shown in TABLE I). At this moment we are certain about that $y[s]$ and $v[s]$ should be written back into $y[T]$ and $v[T]$. But when s is executing its operations with suspicious identity in $[t_1, t_2]$, $y[s]$ and $v[s]$ could also be discarded if s is proved malicious. Therefore $y[T]$ and $v[T]$ are invalid since $y[s]$ and $v[s]$ are created till they are replaced by $y[s]$ and $v[s]$, but these invalid data is not isolated from trustworthy users in DAIS. If they are accessed by trustworthy users, leakage of invalid data might cause damage spreading (In example 1, $y[T]=8$ and $v[T]=8$ are invalid since T_{s1} and T_{s2} commit till they are replaced with $y[s]=13$ and $v[s]=13$, but the trustworthy transactions T_{u2} and T_{u3} both read the invalid $y[T]=8$, when T_{u2} and T_{u3} commit, $x[T]$ and $z[T]$ will be infected invalid too.) According to the above discussion, we can see that DAIS can not satisfy NLID criterion.

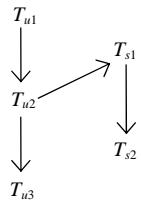


Figure 3. Precedence Graph G

TABLE I. VALUE OF DATA VERSIONS

Data Item	x		y		z		v	
Data Versions	$x[T]$	$x[s]$	$y[T]$	$y[s]$	$z[T]$	$z[s]$	$v[T]$	$v[s]$
5 Transactions Finished	21	—	8	13	18	—	8	13
After Data Version Replacement	21	—	13	DEL	18	—	13	DEL

a. "DEL" means data deletion; b. "—" means data not exists; c. $x=y=z=v=8$ at beginning

III. SQRM METHOD

We assume that all operations of users are trustworthy when the users are trustworthy, suspicious when the users are suspicious and malicious when the users are malicious. The identity transition diagram of user is shown in Figure 4. A trustworthy user can be detected suspicious, and a suspicious user can be proved innocent (trustworthy) or malicious. Furthermore, because suspect detecting method is not the purpose of this paper, we assume that detection of suspicious users is accurate and prompt (In fact, suspicious detection could be simple and efficient in practical applications. For example, when a user logs on system from an unknown address or an abnormal time, this user could be suspicious).

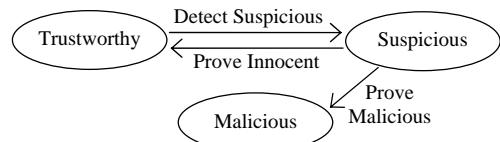


Figure 4. Identity transition diagram of users

In this section, we will formally describe the theoretical concepts and algorithms of SQRM, including data modal, user operation quarantine algorithm and invalid data recovery algorithm. The key points of SQRM are to isolate invalid data from trustworthy users to prevent damage spreading, provide the quarantined extra value of data items to suspicious users for catching results of their work instead of stopping them arbitrarily, and recover the invalid data as soon as possible.

A. Data Model

A database system could be seen as a set of data items, we denote it as $DB=\{x_1, x_2 \dots x_n\}$. There are two value types of data item as shown in definition 1.

Definition 1. Each data item $x_i \in DB$ could have two value types:

- Normal Type value (NT-value). It is maintained by all trustworthy users and denoted as x_i^N . We use $DB^N = \{x_i^N \dots x_j^N\}$ to represent NT-value set.
- Quarantined Type Value (QT-value). It is maintained only by suspicious user who created it and denoted as x_i^Q . The user maintaining x_i^Q is denoted as $owner(x_i^Q)$. We use $DB^Q = \{x_i^Q \dots x_k^Q\}$ to represent QT-value set.

NT-value and QT-value of a data item are transparent to users. If a user submits to accessing a data item x_i successfully, only one value (x_i^N or x_i^Q) of x_i will be accessed. Data accessing measures follow user operation isolation algorithm in Section 3.2. When database is initialized to start service, only NT-value of data items exists, and all data items are trustworthy so as to be able of being accessed by trustworthy users at that time. We give the definition of trustworthy data as follows.

Definition 2. Trustworthy Data. For a data item $x_i \in DB$, if its NT-value x_i^N exists and QT-value x_i^Q does not exist, x_i is a trustworthy data. We use \mathfrak{R} to represent the trustworthy data set.

$$\mathfrak{R} = \{x_i / x_i \in DB \wedge \exists x_i^N \in DB^N \wedge \neg \exists x_i^Q (x_i^Q \in DB^Q)\} \quad (1)$$

However, when suspicious user emerges, invalid data could be produced because of suspicious activities in SUQ-Phase. And when suspicious user is proved innocent or malicious, invalid data will be recovered to be trustworthy finally in IDR-Phase. We will give definitions of invalid data and discuss it in next sections.

B. User Operation Isolation Algorithm

Once a suspicious user is detected, it should be quarantined efficiently, and make sure that suspicious users will be able to continue their work in isolation instead of

being stopped arbitrarily, meanwhile the invalid data will not be accessed by trustworthy users to prevent damage spreading. To achieve this goal, data accessing operations of users should be controlled as shown in User Operation Isolation Algorithm (UOIA). Here, $read(x_i^Q/x_i^N)$, $write(x_i^Q/x_i^N)$ and $create(x_i^Q/x_i^N)$ are the read, write and create operations on NT-value or QT-value of x .

Algorithm 1: UOIA Pseudo Code

Input: User s submit read or write operation on data item x_i
Output: Result of operation (TRUE or FALSE)
Steps:

- 1: **IF** s is trustworthy **THEN**
- 2: **IF** $\neg\exists x_i^Q(x_i^Q \in DB^Q) \wedge \exists x_i^N(x_i^N \in DB^N)$ **THEN**
- 3: execute $read(x_i^N)$ or $write(x_i^N)$;
- 4: **RETURN** TRUE; // user operation succeed
- 5: **END IF**
- 6: **ELSE IF** s is suspicious **THEN**
- 7: **IF** s submit read on x_i **THEN**
- 8: **IF** $\exists x_i^Q(x_i^Q \in DB^Q \wedge owner(x_i^Q) = s)$ **THEN**
- 9: execute $read(x_i^Q)$;
- 10: **RETURN** TRUE;
- 11: **ELSE IF** $\neg\exists x_i^Q(x_i^Q \in DB^Q) \wedge \exists x_i^N(x_i^N \in DB^N)$ **THEN**
- 12: execute $read(x_i^N)$;
- 13: **RETURN** TRUE;
- 14: **END IF**
- 15: **ELSE IF** s submit write on x_i **THEN**
- 16: **IF** $\exists x_i^Q(x_i^Q \in DB^Q \wedge owner(x_i^Q) = s)$ **THEN**
- 17: execute $write(x_i^Q)$;
- 18: **RETURN** TRUE;
- 19: **ELSE IF** $\neg\exists x_i^Q(x_i^Q \in DB^Q) \wedge \exists x_i^N(x_i^N \in DB^N)$ **THEN**
- 20: execute $create(x_i^Q)$; //create QT-value
- 21: set $owner(x_i^Q) = s$; // set ownership of QT-value
- 22: execute $write(x_i^Q)$;
- 23: **RETURN** TRUE;
- 24: **END IF**
- 25: **END IF**
- 26: **RETURN** FALSE; // user operation failed

We can see that in UOIA: a) When a trustworthy user s wants to read or write data item x_i , only if x_i^N exists and x_i^Q does not exist (which means that x_i is trustworthy), the read or write operation on x_i^N will be executed, otherwise it will fail. b) When suspicious user s wants to read x_i , if x^Q owned by s exists, x_i^Q will be returned to s , while if x_i^Q does not exist and x_i^N exists, x_i^N will be returned to s . c) When suspicious user s wants to write x_i , if x_i^Q owned by s exists, the write operation on x_i^Q will be executed, while if x_i^Q doesn't exist and x_i^N exists, x_i^Q will be created, and the write operation on x_i^Q will be executed in the end. Note that the algorithm is based on strict two-phase-locking (2PL) [24] concurrency control protocol with data item locking granularity.

Data accessing of suspicious users could cause trustworthy data to be invalid. For a trustworthy data x , if a suspicious user s submits write operation (UPDATE) on it, according to UOIA, the QT-value x_i^Q owned by s will be created, so x_i^N and x_i^Q will both exist. Due to the suspicious identity of s , x_i^Q is also suspicious. If s is indeed malicious, x_i^Q should be discarded since it is incorrect value of x_i and x_i^N will be identified as correct. Reversely, if s is indeed

trustworthy, x_i^Q is actually the correct value of x_i reversely, while x_i^N is turned to be incorrect and should be replaced with x_i^Q . Obviously, the correct value of data item x_i is undetermined, either x_i^N or x_i^Q is correct. So accessing x_i^N or x_i^Q by trustworthy users could harm trustworthy data and cause damage spreading, leading this kind of data items invalid. Particularly, if s submits a new data creation operation (INSERT) successfully, a particular data item will be created with only QT-value existence, which renders situation similar to above: Suppose that data item x_i with only x_i^Q existence is created by s , if s is indeed trustworthy, x_i will be also trustworthy, but if s is indeed malicious, x_i should be non-existent, so this kind of data item is also uncertain and invalid. Therefore, to isolate invalid data like x_i from trustworthy users is essential to prevent damage spreading. Here we give the definition of invalid data as follows.

Definition 3. Invalid Data: For a data item $x_i \in DB$, if the QT-value x_i^Q exists, x_i is an invalid data. We denote invalid data set of database as \mathfrak{I} , and invalid data set caused by suspicious user s as $IDS(s)$.

$$\mathfrak{I} = \{x_i / x_i \in DB \wedge \exists x_i^Q(x_i^Q \in DB^Q)\} \quad (2)$$

$$IDS(s) = \{x_i / x_i \in DB \wedge \exists x_i^Q(x_i^Q \in DB^Q \wedge owner(x_i^Q) = s)\} \quad (3)$$

Since invalid data is caused by suspicious users, if we use $S = \{s_1, s_2, \dots, s_m\}$ to represent all suspicious users, we can get an equation about invalid data set as follows.

$$\mathfrak{I} = \bigcup_{s_i \in S} IDS(s_i) \quad (4)$$

Lemma 1. $DB = \mathfrak{R} \cup \mathfrak{I}$

Following the definition of \mathfrak{I} and \mathfrak{R} , it is easy to see that Lemma 1 is true.

Lemma 2. UOIA can ensure all invalid data of \mathfrak{I} will be isolated from trustworthy users.

Proof: (Sketch) According to UOIA procedures, for each data item $x_i \in DB$, only if x_i^N exists and x_i^Q does not exist, which means that x_i is trustworthy, x_i can be read or written by trustworthy users. But if x_i^Q exists, the access to data item x_i by any trustworthy users will fail. So Lemma 2 holds.

As known from Lemma 2, all invalid data will be isolated from trustworthy users to prevent damage spreading, and QT-value of data items will be provided to continue the work of suspicious users in isolation. Therefore, work of SUQ-Phase can be accomplished by following UOIA.

C. On-the-fly Invalid Data Recovery Algorithm

Once a suspicious user is proved innocent or malicious, for each invalid data x_i caused by it, the correct value of x_i will be identified, and the IDR-Phase will start. The key points of the recovery measures are as follows: If s is proved innocent, the QT-value of invalid data owned by s is correct to be written back into the corresponding NT-value, if s is proved malicious, the QT-value of invalid data owned by s is incorrect and should be deleted. To implement the above measures, we give an On-the-fly Invalid Data Recovery Algorithm (OIDRA) as shown in Algorithm 2. Here,

$delete(x_i^Q/x_i^N)$ represent delete operation on NT-value or QT-value of data item x_i .

In OIDRA, once a suspicious user s is proved innocent or trustworthy, the performing operations of s will be canceled and s will be stopped from data accessing to begin invalid data recovery. Then, if s is proved innocent, all QT-value of invalid data owned by s is correct, they will be adopted to replace the corresponding NT-value (if NT-value does not exist, it will be created firstly). After that the data accessing authority of s will be resumed. If s is proved malicious, all QT-value owned by s will be deleted. When above procedures finished, all QT-value owned by s will be dropped, and all invalid data of $IDS(s)$ will be recovered to end the IDR-Phase of s . Therefore the work of IDR-Phase can be fulfilled by OIDRA.

Algorithm 2: OIDRA Pseudo Code

Input: a signal that s is proved innocent or malicious
Output: Recovery of invalid data
Steps:

- 1: Cancel all performing operations of s , and stop s from accessing database;
- 2: **IF** s is proved innocent **THEN**
- 3: **FOR EACH** $x_i^Q \in DB^Q \wedge owner(x_i^Q) = s$
- 4: **IF** $\neg \exists x_i^N (x_i^N \in DB^N)$ **THEN**
- 5: execute *create*(x_i^N);
- 6: **END IF**
- 7: set $x_i^N = x_i^Q$;
- 8: execute *delete*(x_i^Q);
- 9: **END FOR**
- 10: Resume data accessing authority for s ;
- 11: **ELSE IF** s is proved malicious **THEN**
- 12: **FOR EACH** $x_i^Q \in DB^Q \wedge owner(x_i^Q) = s$
- 13: execute *delete*(x_i^Q);
- 14: **END FOR**
- 15: **END IF**

Furthermore, since all invalid data is isolated from trustworthy users as known from Lemma 2, OIDRA can performs concurrently with other operations, so OIDRA is an on-the-fly algorithm.

Lemma 3. OIDRA can ensure that all invalid data of \mathfrak{I} will be recovered in the IDR-Phase of suspicious users.

Proof: (*Sketch*) According to OIDRA procedures, once a suspicious user s is proved innocent or malicious, all invalid data of $IDS(s)$ caused by s will be recovered. Since every suspicious user will be proved innocent or malicious finally, each invalid data caused by suspicious users will be recovered in the end. While all invalid data caused by all suspicious users is just the invalid data set \mathfrak{I} as known from definition 2, so Lemma 3 can be satisfied.

Lemma 4. SQRM can satisfy NLID criterion.

Proof: (*Sketch*) According to Lemma 2 and Lemma 3, we can see that all invalid data caused by suspicious users will be isolated from trustworthy users, so damage spreading will be prevented, and all invalid data will be recovered in the IDR-Phase of suspicious users. As a result, the NLID criterion can be satisfied in SQRM.

Therefore, the suspect quarantine and recovery method we proposed in this paper is an effective security mechanism, which can make sure that further damage by

damage spreading will be confined and all invalid data will be recovered.

IV. CONCLUSIONS

In this paper, we presented an effective security mechanism, namely suspect quarantine and recovery method denoted as SQRM, to increase the attack resistance of database vulnerable to suspicious users. We develop a suspicious user quarantine scheme in SQRM that isolates invalid data from trustworthy users to protect databases from any further damage caused by damage spreading, and provides the ability of working continually to suspicious users instead of stopping them arbitrary. At the same time, we propose an on-the-fly invalid data recovery method to repair all the invalid data caused by suspicious users when they are proved innocent or malicious.

There are some future works for SQRM. First, since transactions, which consist of access operations, will be suspended or aborted even if only one invalid data accessed, we will investigate the effect of SQRM on transaction success rate. Second, the availability of database when dealing with the quarantine of suspicious users will be studied. Furthermore, we will concentrate on construction of survival database system which is able to solve the problem of suspicious users by SQRM.

ACKNOWLEDGMENT

Our work is supported by the National Natural Science Foundation of China (60673127), the National 863 High Technology Research and Development Program of China (2007AA01Z404) and the Jiangsu Province Science & Technology Pillar Program (BE2008135).

REFERENCES

- [1] E. Bertino and R. Sandhu, "Database Security-Concepts, Approaches, and Challenges," IEEE Transactions on Dependable and Secure Computing, vol. 2, no. 1, pp. 2-19, Jan.-Mar. 2005, doi:10.1109/TDSC.2005.9.
- [2] P. Ammann, S. Jajodia, and C. D. McCollum, "Surviving information warfare attacks on databases," Proc. 1997 IEEE Symp. Security and Privacy, IEEE Press, May. 1997, pp. 164-174, doi:10.1109/SECPRI.1997.601331.
- [3] P. Liu, "Architectures for Intrusion Tolerant Database Systems," Proc. 18th Annual Computer Security Applications Conference, IEEE Press, Dec. 2002, pp. 311-320, doi:10.1109/CSAC.2002.1176303.
- [4] T. Chiueh and D. Pilania, "Design, implementation, and evaluation of a repairable database management system," Proc. 21th Int. Conference on Data Engineering, IEEE Press, Apr. 2005, pp. 1024-1035, doi:10.1109/ICDE.2005.49.
- [5] P. Liu and J.W. Jing, "The Design and Implementation of a Self-Healing Database System," Journal of Intelligent Information Systems, vol. 23, Nov. 2004, pp. 247-269, doi:10.1023/B:JIIS.00000047394.02448.8d.
- [6] Y. Hu and B. Panda, "A data mining approach for database intrusion detection," Proc. 2004 ACM Symp. Applied Computing, ACM Press, Mar. 2004, pp. 711-716, doi:10.1145/967900.968048.
- [7] M. Vieira and H. Madeira, "Detection of Malicious Transactions in DBMS," Proc. 11th IEEE Int. Symp. Pacific Rim Dependable Computing, IEEE Press, Dec. 2005, pp. 350-357, doi:10.1109/PRDC.2005.31.
- [8] J. Fonseca, M. Vieira, and H. Madeira, "Integrated Intrusion Detection in Databases," Proc. 3rd Latin-American Symp. Dependable

- Computing, Springer-Verlag Press, Sep. 2007, pp. 198-211, doi: 10.1007/978-3-540-75294-3_15.
- [9] A. Kamra, E. Terzi, and E. Bertino, "Detecting anomalous access patterns in relational databases," *The VLDB Journal*, vol. 17, Aug. 2008, pp. 1063-1077, doi:10.1007/s00778-007-0051-4.
- [10] P. Liu and S. Jajodia, "Multi-phase Damage Confinement in Database Systems for Intrusion Tolerance," Proc. 14th IEEE Computer Security Foundations Workshop, IEEE Press, Jun. 2001, pp. 191-205, doi:10.1109/CSFW.2001.930146.
- [11] K. Bai and P. Liu, "A light weighted damage tracking quarantine and recovery scheme for mission-critical database systems," Proc. 17th ACM Conference on Information and knowledge management, ACM Press, Oct. 2008, pp. 1403-1404, doi:10.1145/1458082.1458302.
- [12] K. Bai, M. Yu, and P. Liu, "TRACE: Zero-Down-Time Database Damage Tracking, Quarantine, and Cleansing with Negligible Run-Time Overhead," Proc 13th European Symp. Research in Computer Security, Springer-Verlag Press, Oct. 2008, pp. 161-176, doi:10.1007/978-3-540-88313-5_11.
- [13] M. Yu, W. Zang, and P. Liu, "Database Isolation and Filtering against Data Corruption Attacks," Proc 23rd Annual Computer Security Application Conference, IEEE Press, Dec. 2007, pp. 97-106, doi:10.1109/ACSAC.2007.18.
- [14] K. Bai and P. Liu, "A data damage tracking quarantine and recovery (DTQR) scheme for mission-critical database systems," Proc. 12th Int. Conference Extending Database Technology, ACM Press, Mar. 2009, pp. 720-731, doi:10.1145/1516360.1516443.
- [15] P. Liu, P. Ammann and S. Jajodia, "Rewriting Histories: Recovering From Malicious Transactions," *Distributed and Parallel Databases*, vol. 8, Jan. 2000, pp. 7-40, doi:10.1023/A:1008731200105.
- [16] P. Ammann, S. Jajodia, and P. Liu, "Recovery from Malicious Transactions," *IEEE Transactions Knowledge and Data Engineering*, vol. 14, Sep. 2002, pp. 1167-1185, doi:10.1109/TKDE.2002.1033782.
- [17] T. Chiueh and S. Bajpai, "Accurate and efficient inter-transaction dependency tracking," Proc. 24th Int. Conf. Data Engineering, IEEE Press, Apr. 2008, pp. 1209-1218, doi: 10.1109/ICDE.2008.4497530.
- [18] D. Lomet, Z. Vagena, and R. Barga, "Recovery from "bad" user transactions," Proc. 2006 ACM SIGMOD Int. Conference Management of Data, ACM Press, Jun. 2006, pp. 337-346, doi:10.1145/1142473.1142512.
- [19] R. Yalamanchili and B. Panda, "Transaction Fusion: A Model for Data Recovery from Information Attacks," *Journal of Intelligent Information Systems*, vol. 23, Nov. 2004, pp. 225-245, doi:10.1023/B:JIIS.0000047393.99078.c4.
- [20] K. Bai and P. Liu, "Towards Database Firewall: Mining the Damage Spreading Patterns," Proc. 22nd Annual Computer Security Applications Conference IEEE Press, Dec. 2006, pp. 449-462, doi:10.1109/ACSAC.2006.52.
- [21] S. Jajodia, P. Liu, and C. D. McCollum, "Application-Level Isolation to Cope With Malicious Database Users," Proc. 14th Annual Computer Security Applications Conference, IEEE Press, Dec. 1998, pp. 73-82, doi:10.1109/CSAC.1998.738580.
- [22] P. Liu, "DAIS: A Real-Time Data Attack Isolation System for Commercial Database Applications," Proc. 17th Annual Computer Security Applications Conference, IEEE Press, Dec. 2001, pp. 219-229, doi:10.1109/ACSAC.2001.991538.
- [23] P. Liu, H. Wang, and L.Q. Li, "Real-time data attack isolation for commercial database applications," *Journal of Network and Computer Applications*, vol 29, Nov. 2006, pp. 294-320, doi:10.1016/j.jnca.2005.03.001.
- [24] P. Bernstein, V. Hadzilacos, and N. Goodman, *Concurrency Control and Recovery in Database Systems*. Addison-Wesley, 1987.

Exploring the Essence of an Object-Relational Impedance Mismatch

- A novel technique based on Equivalence in the context of a Framework

Christopher Ireland, David Bowers, Michael Newton, Kevin Waugh

Department of Maths and Computing

The Open University

Milton Keynes, UK

{cji26@student.open.ac.uk, D.S.Bowers@open.ac.uk, M.A.Newton@open.ac.uk, K.G.Waugh@open.ac.uk }

Abstract- During the development of an object-relational application we combine technologies that make use of object and relational artefacts because each is suited to a particular role. However such a combination of technologies gives rise to problems of an object-relational impedance mismatch. In this paper we highlight these problems arise not just because of differences in language or design objective, but because the semantics and data of an object and a relational artefact are not equivalent. We introduce a novel technique based on equivalence, and use this to explore one problem of an object-relational impedance mismatch. We show that strategies for dealing with the problem of identity should not focus on a correspondence between the two identity systems but on a correspondence between the different ways in which the identity of an entity has been represented.

Keywords- object-relational; impedance mismatch; ORIM; silo; equivalence

I. INTRODUCTION

Object and relational technologies have proven popular for the development, respectively, of applications and databases, but there are problems that occur when we attempt to combine them in a software system. Each such problem is commonly referred to as an object-relational impedance mismatch (ORIM) [1].

In [2] we explore problems of an ORIM and conclude that there are four kinds of mismatch (conceptual, representation, emphasis and instance), each reflecting a different abstraction (respectively, concept, language, schema and instance). Our framework (Figure 1) recognises two collections of concepts, each provides the basis for a *silo*. A silo comprises artefacts from an abstraction at each level of our framework. The object silo is the left side of Figure 1 and the relational silo is the right side. A level provides a context for the level below.

Our framework highlights that an object and a relational artefact are based on different conceptual frameworks. At the language level an artefact in a silo is described using a particular language. This language is different between silos, for example Java may be used in the object silo and SQL-92 in the relational silo. At the schema level an artefact is created based on a particular design objective. These objectives differ between silos. For example, the design of a program may focus on efficient processing whereas the design of a database may focus on an efficient data structure.

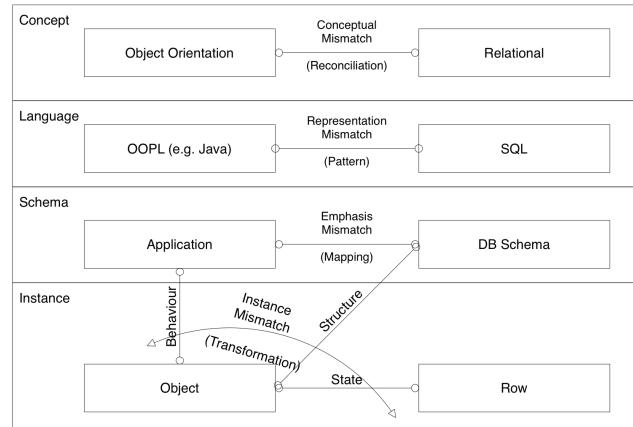


Figure 1. Our Conceptual Framework¹

During the development of an object-relational application, we combine technologies that make use of these artefacts because each is suited to a particular role. In this paper we highlight that problems of an ORIM arise not just because of differences in language or design objective, but because the semantics and data of an object and a relational artefact are not equivalent.

We develop the idea of equivalence in the context of ORIM and our framework. We provide an example based on the concept of identity and find that there is very little in common between object and relational artefacts. We show that current pattern-based strategies to map identity between object and relational artefacts (e.g. Blaha [3], p420 and Keller [4], p21) have focused on mapping the wrong things. They draw a correspondence between two identity systems but these serve to identify different things. We show that a correspondence should be made between the ways the identity of an entity from a universe of discourse has been modelled, because such an identity is common to both representations. Finally we explore the consequences of equivalence in terms of our framework and propose a new silo.

The paper is structured as follows. Section II sets the context for our work. In Section III we describe the

¹ We have chosen to use the label concept rather than paradigm because we understand that a paradigm underpins a conceptual framework. Object and relational are the names of two conceptual frameworks.

schema level of our framework. In Section IV we begin our exploration of equivalence. In Section V we explain our novel idea of equivalence and introduce an equivalence diagram as a mechanism for exploring a problem of an ORIM. In Section VI we provide an example based on a small case study. In Section VII we explore the options for describing an entity and in Section VIII we highlight the consequences of equivalence for our framework. We present a summary and our conclusions in Section IX and describe future work in Section X.

II. PROBLEMS OF AN ORIM

In [2] we catalogued a number of different kinds of problem of an ORIM. These problems arise because there are differences between the artefacts in each silo. Differences are those of data representation, language syntax and semantics, design approach and conceptual framework.

Object-relational mapping (ORM) strategies have been developed to overcome these differences ([3], [4], and [5]). Each such strategy is based on a correspondence between artefacts in the two technologies. At the language level for example, the definition of a class is used as the basis for the definition of a table.

The rationale for such a correspondence may be that artefacts in different silos appear to be the same abstraction because they have the same name. For example we can name a table ORDER and a class ORDER, or a column QUANTITY and an attribute QUANTITY. Using the same name for two artefacts may appear to endow them with the same semantics, but this is a correspondence that is not justified because they are different abstractions.

A strategy may arise because of a perceived need to represent all of the semantics of an object model in a relational database. One such example is the representation of the semantics of a class hierarchy using SQL-92 (a language that has no such explicit semantics [6]). However using SQL-92, it is not necessary to represent all the semantics of a class hierarchy in order to realise the benefits of a class hierarchy in the design of a relational database.

We argue that a singular focus on correspondence between language artefacts is incorrect. The focus should be on the data and semantics of that which is being represented. In particular the way these data and semantics have been represented using those artefacts.

III. THE SCHEMA LEVEL

We start at the schema level of our framework because it relates directly to the work of those involved in the design of an object-relational application. At this level we are concerned with design artefacts that comprise respectively an object-oriented application and a relational database. We consider the design of each to be a form of schema.

At the schema level of our framework, an object model and a relational model describe aspects of a universe of discourse ([7], p2-1). Whilst a schema uses a particular

conceptual framework, language and structure(s) to describe that universe, each schema is a partial representation of the same universe. A universe of discourse therefore provides a point of reference common to both an object and a relational schema. These schemata must be equivalent descriptions of that universe, if we are not to lose information (data and semantics) in a round-trip transformation between an object-oriented application and a relational database. In the next few sections we explore what we mean by an equivalent description at the schema level of our framework.

IV. TWO REPRESENTATIONS OF AN ENTITY

The design of an object-relational application comprises two schemata: one based on the concept of an object and the other on the concept of a relation. Each schema is an abstraction of the same universe of discourse because it is part of the same system. Each schema is also a correct and valid representation of that universe.

The two schemata are also different. Each schema should be based on a collection of concepts, phrased in a particular language and influenced by a design objective. We make the distinction between the formal prescriptive nature of the concepts that underpin the relational model and the relatively descriptive nature of those that underpin an object model. An SQL-92 schema is prescriptive insofar as its language dictates the form of structure into which a representation must fit. An object schema is relatively more descriptive because the semantics and structure of a class are not prescribed in the same way as those of a table. A different person may also produce and therefore influence each schema ([8], p111).

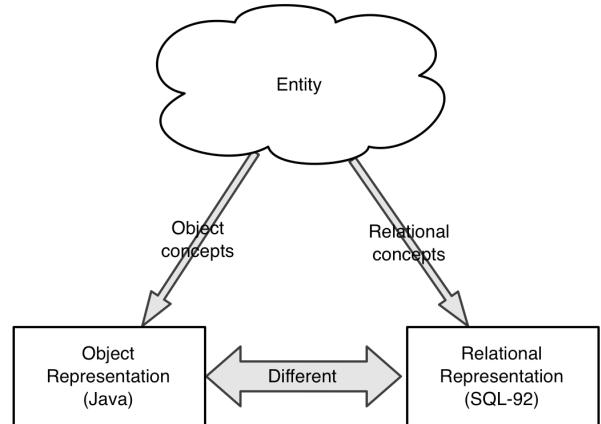


Figure 2. Two Representations of an Entity (type) at the Schema Level

Figure 2 shows an object and a relational representation of the same entity (or entity type) at the schema level. The object representation is formed using artefacts from the Java language. The relational representation is formed using artefacts from SQL-92. We assume that each representation is as complete a representation of the data and semantics of an entity as are possible within a silo.

An entity forms part of a universe of discourse and the description of its data and semantics provides a common

point of reference for both representations. An entity may also be understood as a generalisation of an object and a relational representation. It represents the data and semantics of some thing from a universe of discourse with which we may compare an object and a relational representation.

V. EQUIVALENCE

We consider two representations to be equivalent if they each describe both the same data and the same semantics of an entity. Only those data and semantics that are equivalent can form part of a non-loss transformation between an object and a relational schema. If all the data and semantics of an entity are described in both an object and a relational schema, then none of the data and semantics of that entity should be lost in a round-trip transfer between an application and a database. There may still be differences but these should not impact on the representation of data or semantics. Where there are data or semantics that cannot be preserved in a round-trip transformation between an application and a database, then one schema is able to describe more (or a different subset) of the data or semantics of an entity than the other. Such differences at the schema level are the essence of the kind of object-relational impedance mismatch we label an emphasis mismatch [2].

In both an object and a relational schema one or more artefacts may be used to describe an entity. We use an equivalence diagram to explore differences in the data and semantics of an entity as represented by these artefacts.

An equivalence diagram embodies our notion of equivalence and focuses attention on the essential aspect of Figure 2: that each schema is a description of the same entity. By equivalence at the schema level we mean equivalent descriptions of the same data and semantics of an entity from a universe of discourse.

An equivalence diagram is a Venn diagram comprising two sets. Each set contains the semantics and data of artefacts used to describe an entity in a particular representation. The intersection of these two sets is those data and semantics of an entity that are captured in both representations. These data and semantics will be preserved in a round-trip between an object-oriented application and a relational database.

In Figure 3 the semantics and data of an entity embodied in artefacts used in a schema are represented by a set, drawn as an ellipse. We show two sets: object and relational. The intersection of the two sets represents the data and semantics of an entity common to both schemas. These data and semantics need not be represented in the same way but they are equivalent both to each other and to an idealised representation of entity.

We can use an equivalence diagram in two ways. In the first, we can use equivalence to explore differences of data and semantics between two representations of an entity. We can ask what data and semantics of an entity can be preserved in a round-trip transition from one representation to another. We provide an example in the following sections.

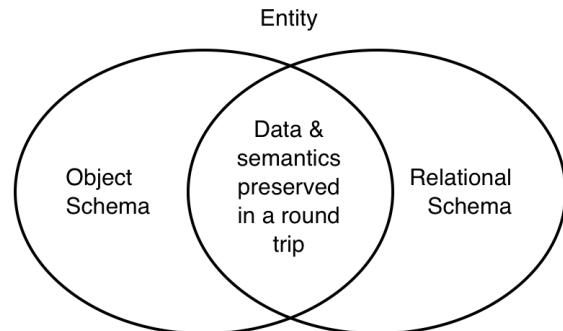


Figure 3. Equivalence between an Object and a Relational representation of an Entity at the Schema Level

In the second, we can use equivalence to improve an ORM strategy. At each level of our framework we can explore the different ways in which the data and semantics of an entity are described by artefacts. At the schema level we consider secondary the artefacts used for the representation of data and semantics of an entity. We describe the consequences for our framework of this use of equivalence in Section VIII.

VI. AN EXAMPLE

In this section we provide an example of the use of an equivalence diagram to explore the identity problem [2]: how do we uniquely identify a collection of data values across both an object and a relational representation?

Figure 4 presents an entity *Equity* taken from a universe of discourse based on an investment bank. *Equity* is a particular financial instrument that represents a share in a company.

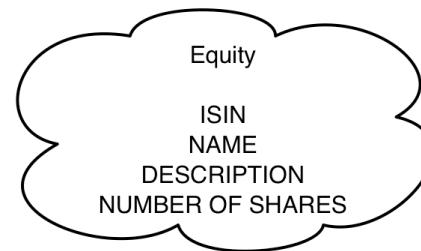


Figure 4. The entity *Equity*.

An equity is identified by an International Securities Identifying Number (ISIN) code. The ISIN code is defined under ISO 6166 and is unique across all financial instruments. The other attributes are self-explanatory.

From Figure 4 we produce an outline class definition shown in Figure 5 and an outline SQL-92 table definition in Figure 6.

An object ID (OID) is implicit and represents the identity of an object. In Java, for example, it is not necessary to define the OID in the definition of the class of which an object is an instance. Hence, there is no mention of an object ID in the definition of class *Equity* in Figure 5.

```

... class Equity
{
    ... ISIN;
    ... NAME;
    ... DESCRIPTION;
    ... NUMBER_OF_SHARES; }

```

Figure 5. The outline of a Java class **Equity** derived from the entity **Equity**

The value of an OID is independent of the value of any of the attributes of an object. For example, although an ISIN is unique in the universe of discourse, the OID of an object of class **Equity** will not be based on the value of the attribute ISIN.

```

create table EQUITY(
    ISIN ... PRIMARY KEY,
    NAME ...,
    DESCRIPTION ...,
    NUMBER_OF_SHARES ...)

```

Figure 6. The outline of an SQL-92 table derived from the entity **Equity**

An OID is unique within the execution space of an object-oriented application. An OID guarantees the uniqueness of an object. Two objects with exactly the same attribute values are different objects if they have a different OID. The identity of an object remains the same regardless of any changes to the value of its attributes. For example, two objects of class **Equity** are different even if they have the same value for the attribute ISIN. In order to prevent this erroneous situation, a constraint must be implemented in a method. Furthermore, changing the value of the attribute ISIN of an object does not change the value of its OID.

The identity of a tuple is the value of all its domains. As such the identity of a tuple is dependent on the value of a domain. In a single relation there cannot be two tuples with the same value for each domain. A primary key of a relation is not necessary for identity but does provide a short-form of reference to a tuple.

At the language level, the semantics of a table are different. A duplicate row is permissible. A primary key enforces uniqueness of a row in a table and restricts the identity of a row to those columns in a key. For example ISIN is the primary key of table EQUITY. There cannot be two rows in this table with the same value for this column. The value of a primary key column in a row should not be changed because this affects the identity of that row.

In Figure 7 we provide an example of an equivalence diagram for the semantics of identity in an object and a relational schema. This shows that there is little in common between the object and relational semantics of identity at the schema level. A row and an object are not the same thing. An OID and the primary key ISIN have little correspondence. The only semantics they share is that each uses identity as a mechanism for ensuring an occurrence is distinct.

These differences are realised at the language level of our framework and above. An OID is not a building block

of an object framework, rather it is a programming necessity introduced at the language level. At the concept level an object is distinct so there is no need for an OID. Similarly, at the language level a primary key uniquely identifies a row in a table. A tuple is distinct by definition. At the concept level therefore we have an object and a tuple, each is unique but they have no common basis for this uniqueness. We have used the levels of our framework to pinpoint the cause of the identity problem. We should not attempt a correspondence between an OID and a primary key because they have no common basis for uniqueness.

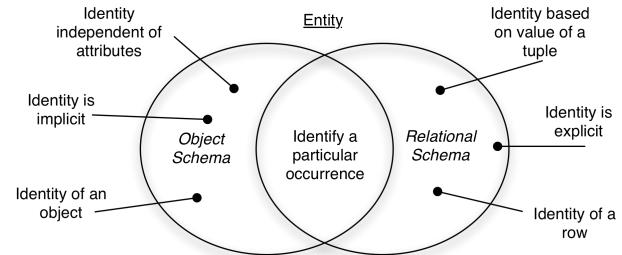


Figure 7. Exploring Identity Between Object and Relational Representations of an Entity

Pattern strategies using SQL-92 (e.g. Blaha [3] and p420, Keller [4], p21) map the semantics of identity between the identity systems employed in an object and a relational schema. For example, they suggest we should introduce a new column into the table EQUITY. This column would store the value of an identity of an object of class **Equity**. Keller [4] suggests using an application-generated identity they call a synthetic identity rather than the actual value of an object ID. Even in our simple example this strategy has shortcomings.

An OID is unique only within the execution space of a single object-oriented program. The correspondence between an ISIN and an OID is only temporary. An OID cannot be used as a primary key because it is not guaranteed to be unique in a database. Even if we extract the value of an ID from an object, that value has no meaning in a database. It also has no semantics in the universe of discourse from which a database schema is derived. A synthetic object ID may somehow be unique across executions but an object and a tuple are different abstractions and such an ID also has no meaning in a universe of discourse. An object ID does not have the semantics necessary to be used as a primary key in a database.

Using equivalence we understand that a mapping between representations at the schema level should be based on correspondence between the mechanisms used to describe the identity of an entity. This mechanism may not be the same as the identity used in each identity system but the identity of an entity is common to both representations. In our example we should make a correspondence between an object and a relational representation of the attribute ISIN of entity **Equity**, because this is the identity of the entity **Equity** and it is common to both representations. We

should not make a correspondence between an object ID and a primary key because these identify different things. A consequence of making a correspondence between these identity systems is that a transformation strategy is then required to form a correspondence between identity values.

VII. DESCRIBING AN ENTITY

The language for describing an entity is an important choice. A description of an entity cannot favour one of the two conceptual frameworks. This would limit the description of the semantics of an entity to those that could be expressed using just one of the frameworks. How then do we describe an entity without favouring one conceptual framework over another?

Dieste [9] describe what they term a generic conceptual model (GCM). The objective of a GCM is to describe knowledge of a requirement in a way that does not determine (what they refer to as) the implementation paradigm. They provide a number of transformations of a GCM into a target conceptual model in a particular implementation paradigm. Whilst the term paradigm was originally intended to describe the set of practices that define a scientific discipline at any particular period of time [10], it has been used in computing as a label for a particular viewpoint. We understand that a paradigm underpins a conceptual framework, and that object and relational are two conceptual frameworks.

The approach of Dieste is set within a software development lifecycle. The objective of a GCM is to delay commitment by providing a description of a universe of discourse independent of an implementation paradigm. Once a choice of implementation paradigm has been made, a GCM is transformed into a model based on a particular collection of conceptual building blocks. A GCM is therefore independent of both an object and a relational conceptual framework.

The language employed in the production of a GCM is a possible candidate for the description of an entity. Unlike an element of a GCM, an entity is not used as the basis for the generation of a representation in a particular conceptual framework. Rather a description of the semantics and data of an entity may be used as the basis for exploring equivalence.

Multi-paradigm Modelling (MPM) is another area in which we may find a candidate for the description of an entity. Multi-paradigm modelling is concerned with “developing a set of concepts and tools to address the challenge of integrating models of different aspects of a software system specified using different formalisms and eventually at different levels of abstraction” [11]. Integrating heterogeneous models is one of the most important challenges of MPM. Amaral [11] notes that “the topic on model composition is of very high interest but one that raises a number of very difficult issues”. Various authors (Jiang et al., Yie et al. and Barroca et al. in [11], p222) have explored dependencies between models, model transformations and language composition. Our framework provides a means to structure an exploration of

these issues. The issue of dependency between models occurs at the schema level of our framework whilst issues of language composition occur at the language level of our framework. Those working in the area of MPM will benefit from our understanding of equivalence because equivalence is essential for the preservation of semantics between models.

VIII. EQUIVALENCE AND OUR FRAMEWORK

We have explored equivalence at the schema level and shown that it may be possible to produce a description of an entity independent of an object and a relational conceptual framework. The concept of an entity is only relevant at the schema level because a schema is a representation of a universe of discourse. In this section we explore the consequences of equivalence in the context of our framework and explain the basis for equivalence at the other levels.

The concept level of our framework provides the context for the language level that in turn provides the context for the schema level. We can use this contextualisation to reflect on the description of an entity at the schema level. The example of identity has highlighted for example, that issues of language influence the semantics of an entity as described in a schema.

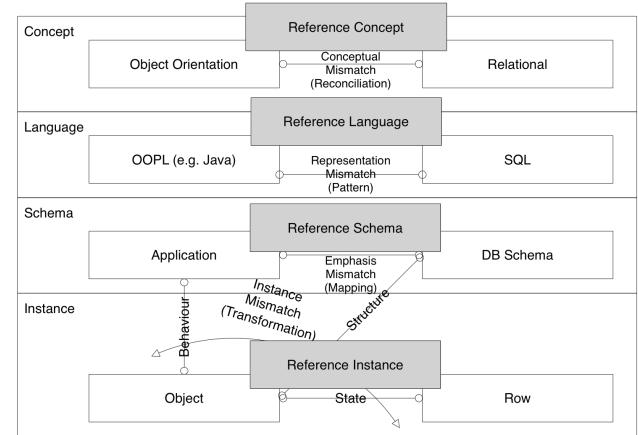


Figure 8. Our Conceptual Framework including the Reference Silo

The description of an entity may be viewed as a generalisation of an object and a relational description. The description of an entity must be phrased in terms of a language that is itself a generalisation of an object and a relational language. The language used to describe an element map ([9], Section 3.1) provides a possible candidate. A conceptual framework that is a generalisation of an object and a relational conceptual framework will underpin the language. We therefore propose a third silo in our framework and we label this the reference silo.

The reference silo (shown down the centre of Figure 8) is currently theoretical and artefacts within it are ideal, but its purpose can be related to the work we describe in Section VII. In this silo there is a reference concept level, a reference language level, a reference schema level and a

reference instance level. Each level provides artefacts for, or influences the description of an entity from a universe of discourse within a reference schema. This description does not need to be perfect, but as a minimum it must be a generalisation of those data and semantics that may be described using an object and a relational artefact.

At each level of our framework within the reference silo we can explore equivalence. We can explore equivalence between the data and semantics of a reference artefact and those data and semantics described in an object and a relational artefact. The data and semantics of a reference artefact described by an object or a relational artefact are shown as a set in an equivalence diagram. Depending on the level of the framework, that set may contain conceptual building blocks, language structures, design representations or data formats.

TABLE I. SOME BUILDING BLOCKS OF THE OBJECT AND THE RELATIONAL CONCEPTUAL FRAMEWORKS

Conceptual Framework	Building Blocks
Object [12]	Object, Class, Association, Method, Attribute, Subclass
Relational [13]	Relation, n-tuple, Domain, Column, Projection, Join, Restriction, Composition, Primary Key

At the concept level of our framework for example, a set comprises the building blocks employed by a conceptual framework. Table I provides an example of some of the building blocks employed by the object and the relational conceptual frameworks. The intersection of the two sets comprises those data and semantics of a reference artefact that are represented by artefacts in both the object and the relational silo.

IX. SUMMARY AND CONCLUSIONS

Problems of an ORIM exist not just because artefacts are described using a different language, but also because an object and a relational representation are based on different conceptual frameworks. This distinction underpins our conceptual framework.

A conceptual framework underpins the language, schema and data used to describe an entity from a universe of discourse. If we are to preserve the data and semantics of an entity from a universe of discourse in a round-trip between an object-oriented application and a relational database, the description of that entity in each schema must be equivalent.

The novel perspective of equivalence facilitates an understanding of an impedance mismatch between an object and a relational artefact. We found at the schema level there is little in common between the semantics of an object and a relational system of identity. ORM strategies have failed to recognise this and instead make a correspondence between identity systems. In order to avoid problems of an ORIM, the correspondence implicit in an ORM strategy should be based on how the data and semantics of the identity of an entity are described in each representation.

Equivalence is not only a schema level concern involving the description of an entity. Reflecting on the contextualisation provided by our framework we introduced the reference silo. This silo comprises the artefacts used to describe an entity at each abstraction.

At each level of our framework we can explore equivalence between an artefact in the reference silo and those in the object and relational silos. Such an exploration will provide further insights into the most appropriate way to address problems of an ORIM.

Whilst the reference silo is still an ideal, we note that there is work in the areas of a GCM and MPM that may lead to the realisation of artefacts in this silo. Our framework will also help those working in the area of MPM.

X. FUTURE WORK

Our technique of equivalence may be used to explore other problems of an ORIM [2]. Such an exploration will demonstrate further our technique, and may result in improvements to other ORM strategies. Finally, further work is required to understand the contribution of our framework and the technique of equivalence, to MPM and an exploration of the issues identified by Amaral [11].

REFERENCES

- [1] G. Copeland and D. Maier: Making Smalltalk a database system. ACM SIGMOD Record 14 (1984) 316-325
- [2] C. Ireland, D. Bowers, M. Newton and K. Waugh: A Classification of Object-Relational Impedance Mismatch. In: Chen, Q., Cuzzocrea, A., Hara, T., Hunt, E., Popescu, M. (eds.): The First International Conference on Advances in Databases, Knowledge and Data Applications, Vol. 1. IEEE Computer Society, Cancun, Mexico (2009) p36-43
- [3] M.R. Blaha, W.J. Premerlani and J.E. Rumbaugh: Relational database design using an object-oriented methodology. Communications of the ACM 31 (1988) 414-427
- [4] W. Keller: Mapping Objects to Tables: A Pattern Language. In: Bushman, F., Riehle, D. (eds.): European Conference on Pattern Languages of Programming Conference (EuroPLoP), Irsee, Germany (1997)
- [5] M.L. Fussell: Foundations of Object Relational Mapping (<http://www.chimu.com/publications/objectRelational/>) (Accessed: 25th September 2007)
- [6] C. Ireland, D. Bowers, M. Newton and K. Waugh: Understanding Object-Relational Mapping: A Framework Based Approach. International Journal On Advances in Software 2 (2009)
- [7] J.J.v. Griethuisen (ed.): Concepts and Terminology for the Conceptual Schema and the Information Base. ISO, New York (1982)
- [8] S.W. Ambler: Agile Database Techniques - Effective Strategies for the Agile Software Developer. Wiley (2003)
- [9] O. Dieste, M. Genero, N. Juristo, J.L. Mate and A.M. Moreno: A conceptual model completely independent of the implementation paradigm. The Journal of Systems and Software 68 (2003) 183-198
- [10] T.S. Khun: The Structure of Scientific Revolutions. The University of Chicago Press, Chicago, IL (1970)
- [11] V. Amaral, C. Hardebolle, G. Karsai, L. Lengyel and T. Levendovszky: Recent Advances in Multi-paradigm Modeling. Models in Software Engineering, Vol. 6002/2010. Springer-Verlag, Berlin (2010) 220-224
- [12] J. Rumbaugh, I. Jacobson and G. Booch: The Unified Modeling Language Reference Manual. Addison Wesley (2005)
- [13] E.F. Codd: A relational model of data for large shared data banks. Communications of the ACM 13 (1970) 377-387

Symbolic Representation and Reasoning for Rectangles with Superposition

Takako Konishi

*Graduate School of Science & Technology
Kwansei Gakuin University
2-1, Gakuen, Sanda, 669-1337, JAPAN
Email: t.konishi@kwansei.ac.jp*

Kazuko Takahashi

*School of Science & Technology
Kwansei Gakuin University
2-1, Gakuen, Sanda, 669-1337, JAPAN
Email: ktaka@kwansei.ac.jp*

Abstract—This paper discusses the superposition of qualitative rectangles so that some parts are visible and other parts are hidden based on the user's requirements. Qualitative rectangles are rectangles whose size and edge ratios are not fixed. We investigate the conditions under which such a superposition succeeds as well as the manner in which such superposition occurs. We also show an algorithm for superposing a multiple number of qualitative rectangles. It is applicable to construct qualitative spatial database for multiple window placement systems.

Keywords-qualitative knowledge representation; rectangle packing; spatial database.

I. INTRODUCTION

This work was inspired by an issue which occurs during multiple window placement. When working on PCs, we often open multiple windows in a superposed manner within the restricted space of a monitor. At that time, we seldom need all the content displayed in the windows; rather, we pick up the necessary portions by frequent use of mouse operations, such as clicking or dragging. Efficient placement of windows such that only the necessary parts are visible could serve as a useful tool to reduce our annoying mouse operations. To achieve this, we should specify the parts of each window to be visible and find the superposition of the windows that satisfies the specification as well as the simple positional relationship on the two-dimensional plane. In general, efficient placement of objects has been studied as a type of packing problem to which an optimal solution is searched [1]. Much work has been undertaken in several application areas, such as VLSI design [2], label-placement problems [3], [4] and more. In these studies, the problem of how multiple objects are located in a two-dimensional plane has been approached under circumstances not involving superposition. To the best of the authors' knowledge, no study has been performed on the location of objects with superposition.

In this study, we discuss rectangle placement with superposition. We treat rectangles using qualitative representation: their sizes and the ratios of their edges are unfixed. In each rectangle, the desired visible part of a rectangle is specified.

We discuss a manner of superposing them so that all desired visible parts are in the foreground and all desired hidden parts are in the background. Figure 1 illustrates several examples. Assume that three rectangles A, B, and C are given with a requirement of visibility specified by a user. The white indicates the parts that one wants to be visible, and the black indicates the parts that one wants to be hidden. In this figure, (a), (b), and (c) are successful cases, whereas (d) is not. In (c), first reduce B's width to fit the vertically-long-size subpart of the black part of A, then C is put on the black part in the lower left part of the resultant figure. In (d), visible black parts remain after superposing A and B. In this paper, we show how we evaluate the success of superposition and placement in these cases.

Considering multiple window placement, a window is divided into several frames in most application software and their dividing patterns are restricted. An automatic window placement can be accomplished by the following mechanism: add the attribute value on visibility to each frame of each window, store in the database the list of pairs of a combination of multiple windows with the attribute values and their best placement; retrieve the best placement from the database for the combination of windows on their invocation, and display them. Attribute value on each frame can be decided through learning from lots of examples, however, this issue is beyond the scope of this paper. Here, we assume that it is given in advance and discuss reasoning about the best placement for a given combination of windows.

We take a qualitative approach. One reason for this is that it enables symbolic handling of objects. In general, spatial data can be inconveniently large to store and handle. Symbolic handling reduces this computational complexity. Another reason is that it is enough to know the relative positional relationship of objects on a two-dimensional plane and their foreground/background relationship, ignoring the exact size or position of each object. Such an idea is considered to be a type of qualitative spatial reasoning (QSR) in the field of artificial intelligence [5], [6], [7], [8].

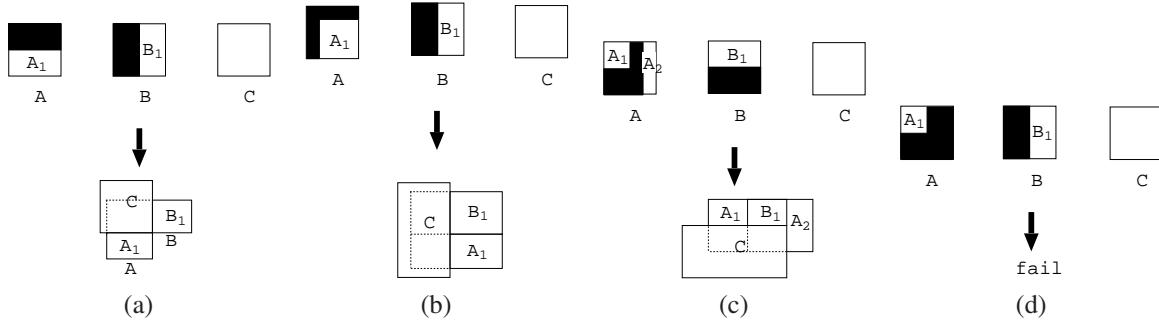


Figure 1. Examples of superposing rectangles

There are several studies on qualitative spatial database. For example, Wang and Liu showed an application of QSR to geospatial semantic web by constructing qualitative spatial database, in which objects and their qualitative relations are stored instead of coordinates, from the Geography Markup Language (GML) [9]. Santos and Amaral proposed an approach to make qualitative database, by introducing qualitative identifier such as direction and relative distance, and apply it to data mining [10]. Although these studies showed effectiveness of qualitative spatial database, further studies are required. This paper aims at enlarging possible application areas of qualitative spatial database.

This paper is organized as follows. In Section II, we define the target object and describe its qualitative representation. In Section III, we describe the operations of superposing a pair of qualitative rectangles, and discuss reasoning about superposition. In Section IV, we discuss the result of superposition and show an algorithm for superposing multiple qualitative rectangles. Finally, in Section V, we show the conclusion.

II. DESCRIPTION

We call a superposing entity *a unit*. A unit is divided into WHITE which should be visible, and BLACK, which should be hidden. BLACK is divided into a *core region* and a *non-core region*, which will be defined later. The outer side of a unit is called GRAY. The length of edges and the ratios of a unit and of each region are unfixed. On the other hand, the orientation of a unit should be fixed. We only use rectangles situated in an upright position and do not consider those in an inclined orientation. These means that (a) and (b) in Figure 2 are regarded as equivalent, while (a) and (c) are regarded as different.

Each connected WHITE is called *a white region*, the core region and each connected non-core region are called *black regions*, and GRAY is called *a gray region*. The white, black, and gray regions have attribute values related to visibility, denoted by 'w,' 'b,' and 'g.' 'w' and 'b' denote that the region should be visible and hidden, respectively. 'g' denotes there is no requirement with respect to visibility.

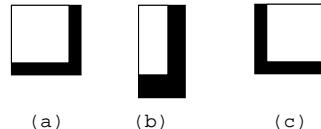


Figure 2. Qualitative rectangles

Considering a structure of frames of WEB pages or a style of dividing a window into sub-windows in many applications, we restrict the type of unit to those in Figure 4. Any unit can be defined as a qualitative rectangle obtained by the following operation that fits black plates into a white rectangle. Conversely, a qualitative rectangle obtained by this operation is only the units shown in Figure 4. Let $a * b$ represent a size of a unit whose length is a and height is b . Consider the two black plates whose sizes are $x * b$ ($0 \leq x \leq a$) and $a * y$ ($0 \leq y \leq b$). Fit these plates into a white rectangle while preserving their orientations in either of the following manners. Symbols enclosed in parentheses denote names of unit types.

- (0) No plate is fit (W).
 - (1) Only one of the plates is fit (B, I1, I2).
 - (2) Both plates are fit (L1, T1, PLUS).
 - (3) Extend L1 and T1, respectively, where the white region is added to the part on which the edge of a size a or b is connected to the outer part (L2, T2).

Definition 1. *The unit obtained in this manner is said to be valid.*

Types I1 and I2 are called *straight-plate-units*. Types L1, L2, T1, T2, and PLUS are called *cross-plates-units*. All units of the same type are called *a pattern*.

For all units other than the W-type unit, the *core region* is defined. For B-type and straight-plate-units, the core region is defined as the entire **BLACK**. For cross-plates-units, the core region is defined as the intersection of the two plates, and the region not included in the core region is called the *non-core region* (Figure 3).

In a valid unit, all white regions are convex, and there

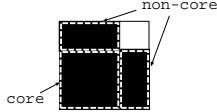


Figure 3. Core region and non-core region of cross-plates-unit

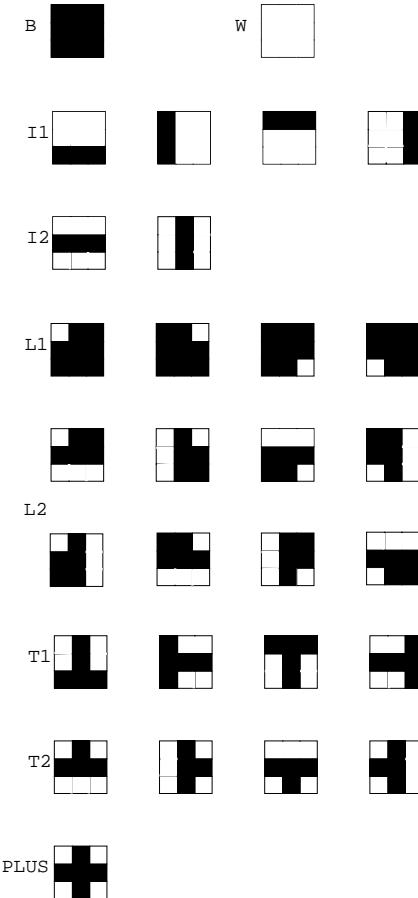


Figure 4. All units

exists only one connected BLACK.

We denote the core region of a unit X by $Core_X$. The valid unit can be uniquely represented as a quadruple of attribute values composed of $Core_X$'s upper region, right region, lower region, and left region. We call this representation *a representation for a unit*. For example, the representation for a unit in Figure 3 is $\langle b, b, g, g \rangle$, since the core region has black regions in its upper side and right side, whereas it is connected to the outside in its lower side and left side. Note that the positional relationships of regions are preserved even if the size of a unit is changed.

Let V, R and $TYPE$ indicate a set of attribute values, a set of representations for units, and a set of types, that is:

$$V = \{b, w, g\}$$

$$R = \{\langle r_1, r_2, r_3, r_4 \rangle \mid r_i \in V (i = 1, \dots, 4)\}$$

$TYPE = \{'B', 'W', 'I1', 'I2', 'L1', 'L2', 'T1', 'T2', 'PLUS'\}$
Then, the function $type$ that defines the type for a representation $r \in R$ is defined as follows.

$type : R \rightarrow TYPE$	
$type(\langle g, g, g, g \rangle)$	= 'B'
$type(\langle w, w, w, w \rangle)$	= 'W'
$type(\langle w, g, g, g \rangle)$	= 'I1'
$type(\langle w, g, w, g \rangle)$	= 'I2'
$type(\langle b, g, g, b \rangle)$	= 'L1'
$type(\langle b, g, w, b \rangle)$	= 'L2'
$type(\langle b, b, g, b \rangle)$	= 'T1'
$type(\langle b, b, w, b \rangle)$	= 'T2'
$type(\langle b, b, b, b \rangle)$	= 'PLUS'

Note that type W is defined with the assumption that there exists a tiny core region surrounded by white regions, as $Core_X$ does not exist.

The projections from $r \in R$ to its elements are defined as follows.

$$up/dn/lr/rt : R \rightarrow V$$

Let r be $\langle r_1, r_2, r_3, r_4 \rangle$.

$$\begin{aligned} up(r) &= r_1 \\ rt(r) &= r_2 \\ dn(r) &= r_3 \\ lr(r) &= r_4 \end{aligned}$$

Moreover, the function $rotate(r)$ that denotes a $\pi/2$ clockwise rotation of a unit r and the function $symm(r)$ that denotes a symmetric transformation of a unit r are defined as follows:

Let r be $\langle r_1, r_2, r_3, r_4 \rangle$.

$$rotate : R \rightarrow R$$

$$rotate(\langle r_1, r_2, r_3, r_4 \rangle) = \langle r_2, r_3, r_4, r_1 \rangle$$

$$symm : R \rightarrow R$$

$$symm(\langle r_1, r_2, r_3, r_4 \rangle) = \langle r_1, r_4, r_3, r_2 \rangle$$

Note that $type(r) = type(rotate(r))$ and $type(r) = type(symm(r))$ hold.

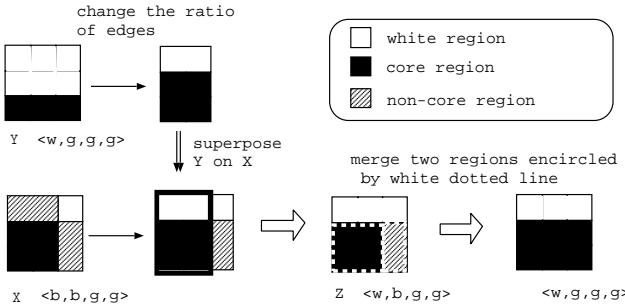
III. REASONING ABOUT SUPERPOSITION

A. The principle

When n ($n \geq 3$) units are given, we consider the manner of their superposition in which all white regions are visible and all black regions are hidden.

Here, we place units sequentially. k -th unit ($n \geq k \geq 2$) should be placed on the figure composed of $k - 1$ units so that at least one region of the former is placed on at least one region of the latter. That is, we do not consider the placement in which, after two units are placed in a disconnected manner, the third unit is placed onto the black region of the two rectangles simultaneously. Thus, there should be at least one W-type unit, and only one connected rectangular BLACK should be visible when superposition of $n - 1$ units is completed.

Here, we assume that there is one W-type unit. When more than one W-type unit exists, the scenario can be considered similarly.

Figure 5. The case in which *merge* is necessary

B. Superposing the core regions

Definition 2. Suppose that a unit X and a straight-plate-unit Y are given. Let Core_X and Core_Y be the core regions of X and Y , respectively. The superposition in which Core_Y is placed exactly on Core_X is called *puton* operation.

Let Z be the resultant figure of *puton*, and let Core_Z be the superposed region of Core_X and Core_Y . We extend a representation for a unit to be available as a representation for Z . A representation for Z is a quadruple of the attribute values of visible regions surrounding Core_Z .

First, we compute the attribute values of the regions around Core_Z . We define the function *on*, which computes the attribute value of the visible region when one region is placed exactly on another region, from those of the two regions.

$$\begin{aligned} \text{on} : V \times V &\rightarrow V \cup \{\text{U}\} \\ \text{on}(b, b) &= b \\ \text{on}(b, w) &= w \\ \text{on}(w, w) &= \text{U} \\ \text{on}(w, b) &= \text{U} \\ \text{on}(g, v) &= v \text{ where } v \in V \\ \text{on}(v, g) &= v \text{ where } v \in V \end{aligned}$$

' U ' means that the operation is undefined for that case.

It sometimes occurs that X 's black regions are visible in Z . If they are connected with Core_Z by lines, it is necessary to merge them to define the merged region as new Core_Z . For example, in Figure 5, X and Y are represented as $\langle b, b, g, g \rangle$ and $\langle w, g, g, g \rangle$, respectively. When we place Y on X such that Core_Y is placed on Core_X , the resultant figure Z is represented as $\langle w, b, g, g \rangle$. X 's non-core region is visible, which is connected with Core_Z by a line. Then, this region is merged with Core_Z . This function *merge* is defined as follows.

Let $r = \langle r_1, r_2, r_3, r_4 \rangle$. If it satisfies (c1), then *merge*(r) succeeds. When *merge* succeeds, its value is defined as follows.

$$(c1) \quad \bigwedge_{i=1, \dots, 4} (r_i \neq \text{U}).$$

$$\text{merge} : R \rightarrow R$$

$$\text{merge}(r) =$$

$$\left\{ \begin{array}{ll} \langle g, r_2, g, r_4 \rangle & \text{if } r_1 = b \wedge r_2 \neq b \wedge r_3 = b \wedge r_4 \neq b \\ \langle r_1, g, r_3, g \rangle & \text{if } r_1 \neq b \wedge r_2 = b \wedge r_3 \neq b \wedge r_4 = b \\ \langle g, r_2, r_3, r_4 \rangle & \text{if } r_1 = b \wedge r_2 \neq b \wedge r_3 \neq b \wedge r_4 \neq b \\ \langle r_1, g, r_3, r_4 \rangle & \text{if } r_1 \neq b \wedge r_2 = b \wedge r_3 \neq b \wedge r_4 \neq b \\ \langle r_1, r_2, g, r_4 \rangle & \text{if } r_1 \neq b \wedge r_2 \neq b \wedge r_3 = b \wedge r_4 \neq b \\ \langle r_1, r_2, r_3, g \rangle & \text{if } r_1 \neq b \wedge r_2 \neq b \wedge r_3 \neq b \wedge r_4 = b \\ \langle r_1, r_2, r_3, r_4 \rangle & \text{otherwise} \end{array} \right.$$

Success of *puton* operation

For representations $r = \langle r_1, r_2, r_3, r_4 \rangle$ and $r' = \langle r'_1, r'_2, r'_3, r'_4 \rangle$, if $\text{merge}(\text{on}(r_1, r'_1), \text{on}(r_2, r'_2), \text{on}(r_3, r'_3), \text{on}(r_4, r'_4))$ succeeds, *puton* succeeds and its value is defined as follows.

$$\text{puton} : R \times R \rightarrow R$$

$$\text{puton}(r, r') =$$

$$\text{merge}(\text{on}(r_1, r'_1), \text{on}(r_2, r'_2), \text{on}(r_3, r'_3), \text{on}(r_4, r'_4))$$

The following property clearly holds due to the definition of *puton*.

Theorem 3. If the *puton* operation succeeds, Z 's BLACK is connected, and all of its white regions are convex.

Next, consider that we superpose the third unit on Z . There are two necessary conditions for this operation to succeed. First, if Z 's BLACK is not rectangular, superposing the W-type on Z will not succeed. Second, if Z is not valid, continuous superposition cannot be considered. We describe how to verify these two conditions.

Shape verification of BLACK

Let r be a representation for Z . If r satisfies both (c2) and (c3), then Z 's BLACK is rectangular.

$$(c2) \quad (\text{up}(r) = b \vee \text{dn}(r) = b) \rightarrow (\text{lt}(r) \neq b \wedge \text{rt}(r) \neq b)$$

$$(c3) \quad (\text{lt}(r) = b \vee \text{rt}(r) = b) \rightarrow (\text{up}(r) \neq b \wedge \text{dn}(r) \neq b)$$

Definition 4. For a figure Z obtained by superposing n ($n \geq 2$) units, if there exists only one connected BLACK that is visible and rectangular, then Z is said to be effective.

Shape verification of the whole figure

Let $r = \langle r_1, r_2, r_3, r_4 \rangle$ and $r' = \langle r'_1, r'_2, r'_3, r'_4 \rangle$ be representations for units X and Y , respectively. For the entire shape of Z to be rectangular, the white region of Y should not be placed on GRAY of X . Therefore, if r and r' satisfy (c4), then the shape of Z is a rectangle.

$$(c4) \quad \text{If there does not exist } i \ (1 \leq i \leq 4) \text{ such that } r_i = g, r'_i = w.$$

C. Superposition by embedding

We can consider another superposition operation of *embed*.

Definition 5. If we place the whole unit on the entirety or on a part of BLACK of the other unit, this operation is called an *embed* operation.

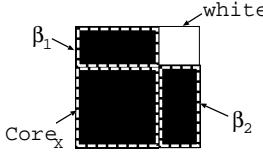


Figure 6. The regions to be hidden in L1

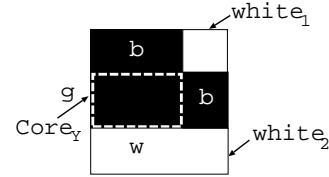


Figure 7. Representation of locations of white regions

IV. RESULT OF SUPERPOSITION

In Definition 2, we defined *puton* operation for a superposition of a straight-plate-unit and a unit. In this section, we extend this operation for any pair of unit types. And we discuss the effectiveness and validity of the resulting figures of the operation *puton* and *embed* for each pair of unit types.

A. Superposition on B/W type

Assume that we superpose some unit on the B-type. The resultant figure is effective if and only if we superpose the straight-plate-unit, and it is valid for any type.

On the other hand, it is impossible to place any unit on the W-type.

B. Superposition of straight-plate-units

Assume that we superpose the straight-plate-unit on the straight-plate-unit. The resultant figure obtained by the *puton* operation is not always valid, as its entire shape may not be a rectangle. The resultant figure obtained by the *embed* operation is not always valid, as a white region may not be convex. On both operations, the resultant figure is effective.

C. Superposition of the straight-plate-unit on the cross-plates-unit

Assume that we superpose the straight-plate-unit on the cross-plates-unit. The resultant figure obtained by any operation is not always valid and not always effective. However, the following property holds.

Theorem 6. *When we superpose the straight-plate-unit on the cross-plates-unit, the effective figure cannot be obtained by any operation other than the puton operation.*

Proof:

Consider the *puton* operation that places a straight-plate-unit Y on an L1-type unit X shown in Figure 6. In this case, BLACK is divided into three regions: one core region $Core_X$ and two non-core regions β_1 and β_2 . Let $Core_Y$ be Y 's core region.

One or two of the $Core_X$, β_1 , β_2 should be hidden so that the resultant superposed figure is effective.

(i) Only one region is hidden.

If only $Core_X$ is hidden, β_1 and β_2 , which are disconnected, are visible. Therefore, the result is not effective. If only β_1 is hidden, $Core_X$, β_2 and $Core_Y$ are visible

in the resultant figure. Considering the relative position of $Core_X$, β_1 and β_2 , it is impossible to make a rectangle by merging $Core_X$, β_2 and $Core_Y$ and to hide β_1 at the same time. Therefore, the result is not effective. Similarly, the result is not effective if only β_2 is hidden.

(ii) Two regions are hidden.

Since β_1 and β_2 are disconnected, they are not simultaneously hidden by a single unit. If both $Core_X$ and β_1 are hidden, β_2 and $Core_Y$ are visible. We must place Y 's regions onto both $Core_X$ and β_1 to make them hidden. Moreover, we must make a rectangle by merging β_2 and $Core_Y$. The only place where $Core_Y$ should be placed to satisfy both conditions is $Core_X$, and this placement is identical to the *puton* operation.

Based on the above analysis, the resultant figure is not effective by operations other than the *puton* operation.

Other cases can be similarly proven. \square

D. Superposition of the cross-plates-unit on any type

Assume that we place the cross-plates-unit on any type of unit.

In this case, the resultant figure is always ineffective with any operation. However, the *puton* operation succeeds for several cases.

In general, when the *puton* operation is performed on X and Y , WHITE should not be placed on X 's WHITE. When Y is a cross-plates-unit, we have to consider its white region located in the inclined orientation from $Core_Y$. The location of white region is represented as the occurrence either of b in adjacent elements or of b and w in adjacent elements in the representation for Y . For example, a representation for a unit in Figure 7 is $\langle b, b, w, g \rangle$. The sequence b, b represents the location of $white_1$, the upper left of $Core_Y$, and the sequence b, w represents that of $white_2$, the lower. Therefore, the condition on WHITE can be represented as (c5).

- (c5) Let $\langle r_1, r_2, r_3, r_4 \rangle$ and $\langle r'_1, r'_2, r'_3, r'_4 \rangle$ be representations for X and Y , respectively. There exists some i ($i = 1, \dots, 4$) such that both $r_i = r'_i \neq g$ and $r_{i+1} = r'_{i+1} \neq g$, where r_5 is regarded as r_1 .

Success of extended *puton* operation

In any pair of units X and Y , if (c1) and (c5) are satisfied, the *puton* operation succeeds.

On the other hand, we can get valid figures by the *embed* operation in some cases. Table I shows the result of the

fg \ bk	L1	L2	T1	T2	PLUS
I1	L1 L2	L1 L2	T1 T2	T1 T2	U*
I2	U	U	T1	T2	U
L1	L1	L2 T2	T1	T2	U*
L2	L2	L2	U*	U*	U
T1	T1	T2	T1	T2	U*
T2	U	U	T2	T2	U
PLUS	U	U	PLUS	U	PLUS

Table I
RESULT OF *embed* FOR CROSS-PLATES-UNITS

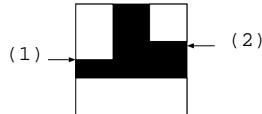


Figure 8. U*: Invalid example

embed operation. In this table, the row shows the unit in the background, and the column shows the unit in the foreground. We show the obtained types when the *embed* operation succeeds for each pair of patterns of each type. U means there is no solution. In case of U*, it appears to be successful at first glance, but there is no solution. For example, Figure 8 shows the resultant figure obtained by the operation of *embed* for L2 on T1. It is not valid because it is impossible to align line (1) and line (2).

E. Algorithm for multiple units superposition

Let Ω be a finite set of valid units where $|\Omega| \geq 2$, and ω is a W-type unit. Then, an algorithm for superposing the units in Ω is shown below.

In the following algorithm, superposition indicates either the *puton* or *embed* operation.

Let $\bar{\Omega}$ be a sequence obtained by setting the elements of Ω in an arbitrary order.

- (1) Let X be a head element X_1 of $\bar{\Omega}$, and set $\bar{\Omega} = \bar{\Omega} - \{X_1\}$.
- (2) Let Y be a head element X_2 of $\bar{\Omega}$, and set $\bar{\Omega} = \bar{\Omega} - \{X_2\}$.
- (3) Let Z be the resulting figure of superposing Y on X .
- (4) If $\bar{\Omega} = \emptyset$,
 - if Z is effective, then success
 - else failure
- else
 - if Z is valid, then set $X = Z$, and goto (2)
 - else failure.

If there exists a sequence $\bar{\Omega}$ that succeeds in this procedure, then the superposition of $\Omega \cup \{\omega\}$ succeeds.

V. CONCLUSION

We have discussed superposition of a pair of units and investigated the conditions that satisfy the result where all

white regions are visible while all black regions are hidden in the resultant figure when visibility is specified by a user.

- A pair of straight-plate-units always produces an effective solution either by the *puton* operation or the *embed* operation.
- The straight-plate-unit on cross-plates-unit can produce an effective solution only by the *puton* operation.
- The cross-plates-unit on any type can produce no effective solution.

As for the last case, we have shown which pairs can generate valid solutions.

We also show an algorithm for superposing a set of units. This is the first study to address object placement with superposition.

Qualitative approach enables the reduction of computational complexity and provides intelligent reasoning by symbolic treatment of spatial data. Although it is effective on spatial database, there have been few works so far. Our contribution is to enlarge possible application areas of qualitative spatial database by showing qualitative representation and reasoning to construct database for multiple window placement systems.

In the future, we are considering weakening the conditions of the unit, such as allowing disconnected black regions.

REFERENCES

- [1] G. Birgin, R. D. Lobato, and R. Morabito, "An effective recursive partitioning approach for the packing of identical rectangles in a rectangle," *Journal of the Operational Research Society*, vol. 61, pp. 306-320, 2010.
- [2] A. S. Lapaugh, "Layout algorithm for VLSI design," *ACM Computing Surveys*, vol. 28, no. 1, pp. 59-61, 1996.
- [3] H. Freeman, "Computer name placement," in *Geographical Information Systems 1*, D. J. Maguire, M. F. Goodchild, and D. W. Rhind, Eds. John Wiley, 1991, pp. 449-460.
- [4] J. Li, C. Plaisant, and B. Shneiderman, "Data object and label placement for information abundant visualizations," in *Proceedings of the Workshop of New Paradigms Information Visualization and Manipulation (NPIV98)*, 1998, pp. 41-48.
- [5] M. Aliello, I. E. Pratt-Hartmann, and J. F. A. K. Van Benthem, Eds., *Handbook of Spatial Logics*. Springer-Verlag, 2007.
- [6] A. Cohn and S. Hazarika, "Qualitative spatial representation and reasoning: an overview," *Fundamental Informaticae*, vol. 46, no. 1, pp. 1-29, 2001.
- [7] M. Egenhofer and R. Franzosa, "On the equivalence of topological relations," *International Journal of Geographical Information Systems*, vol. 9, no. 2, pp. 133-152, 1995.
- [8] S. Kumokawa and K. Takahashi, "Rectangle reasoning: a qualitative spatial reasoning with superposition," in *Proceedings of 23rd Florida Artificial Intelligence Research Society Conference (FLAIRS23)*, 2010, pp. 150-151.
- [9] S. Wang and D. Liu, "Qualitative spatial relation database for semantic web," in *First Asian Semantic Web Conference (ASWC)*, 2006, pp. 387-399.
- [10] M. Santos and L. Amaral, "Geo-spatial data mining in the analysis of a demographic database," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 9, no. 5, pp. 374-384, 2005.

New fuzzy multi-class method to train SVM classifier

Taoufik Guernine

Département d'Informatique, Faculté de Sciences
Université de Sherbrooke
Sherbrooke, Canada
Taoufik.guernine@usherbrooke.ca

Kacem Zeroual

Département d'Informatique, Faculté de Sciences
Université de Sherbrooke
Sherbrooke, Canada
Kacem.zeroual@usherbrooke.ca

Abstract—In this paper we present a new classification method based on Support Vector Machine (SVM) to treat multi-class problems. In the context of multi-class problems, we have to separate large number of classes. SVM becomes an important machine learning tool to handle multi-class problems. Usually, SVM classifiers are implemented to deal with binary classification problems. In order to handle multi-class problems, we present a new method that builds dynamically a hierarchical structure from training data. Our multi-class method is based on three main concepts : Hierarchical classification, Fuzzy logic and SVM. We combine multiple binary SVMs to solve multi-class problems. The proposed method divides the original problem into sub-problems in order to reduce its complexity.

Keywords-Classification; SVM; Fuzzy logic;

I. INTRODUCTION

Solving multi-class problems with high performance is a challenging problem because there is an important increasing processing of data in databases. Until now, multi-class problems remain among the primary worry in the field of classification. Furthermore, the manual classification is not able to keep up with the growth of data. An automatic classification becomes necessary. Many machine learning methods and statistical techniques has been proposed : Decision trees [1], Nearest neighbor classifiers [2], Bayesian models [3] and Support Vector Machine [4].

Unlike the other classifiers, SVM classifiers find an optimal hyperplane maximizing the marge between two classes. Generally, SVM is used for binary classification but its extension to multi-class problems remains an open research topic [5]. There are two techniques for extending SVM to multi-class problems. The first technique consists in resolving optimization problems where the whole training data set is used [6]. This technique requires huge time to train all the data set. The second technique consists in constructing binary classifiers from the root until leaves [7]. The original problem is subdivided into simple binary sub-problems. Each sub-problem contains a small portion of data and is less complex than the original problem. In this paper, we are interested in subdividing the original problem into binary sub-problems. We propose a new classification method based on SVM to treat multi-class problems. The proposed method uses a fuzzy hierarchical structure to extract relationships

between objects. It introduces the transitive closure measure to discover fuzzy similarity between objects. Training data set of SVM obtained a priori by the transitive closure Min-Max assures discriminating between positive and negative classes. Introducing membership values extracted from transitive closure matrix to SVM optimization problem allows high performance.

The remainder of this paper is organized as follows. In section II, we provide an overview of related works. In section III, we give a brief review of SVM. In section IV, we describe the fuzzy hierarchical classification method. In section V, we present our experimental results. Our future research works are presented in section VI.

II. RELATED WORKS

The most important issue in multi-class problems is the existence of confusion classes [8]. The hierarchical structure is among techniques used to solve the confusion classes. The multi-class problems based on SVM is mainly related to hierarchical multi-class pattern recognition problems. Most of recent works used hierarchical structure to address the classification task. In [9], they proposed a new classification algorithm based on a hierarchical structure. The algorithm consists of the following stages : (i) generating category information tree (ii) hierarchical feature propagation (iii) feature selection of category information and (iv) single path traversal. The proposed hierarchical classification system allows adding new categories as required, organizing the web pages into a tree structure and classifying web pages by searching through only one path of the tree structure. In [10], authors explore a hierarchical classification to classify heterogenous collections of the web content. They used hierarchical structure in order to distinguish a second level category from other categories within the same top level. They introduced SVM at each level to obtain a hierarchy. In [11], authors added fuzzy membership values to each input data and reformulate the SVM optimization problem. The membership values make more contribution in the classification process. The proposed fuzzy SVM can solve different kinds of multi-class problems. In [12], the fuzzy set theory is introduced in the classifying module. The authors proposed a One-against-all fuzzy SVM (OAA-

FSVM) classifier to implement a multi-class classification system. The empirical results obtained by the proposed system show that OAA-FSVM method performs better than OAA-SVM method.

III. SUPPORT VECTOR MACHINE

In this section we give a brief review of Support Vector Machine. We present respectively binary and multi-class classification.

A. Binary classification

Generally, SVM classifiers are designed to solve binary classification problem [13]. It consists in minimizing the empirical classification error and finding optimal hyperplane with large margin [14]. Suppose a data set $(x_i, y_i) : (i = 1, \dots, n)$, where x_i corresponds to the attribute set for the i^{th} element. Let $y_i \in \{-1, +1\}$ be a labelled class. The optimal hyperplane can be found by minimizing the margin w in equation III-A :

$$(P) = \begin{cases} \text{Min} \frac{1}{2} \|w\|^2 \\ y_i(wx_i + b) \geq 1 : i \in 1, n : \forall x \in R^n \end{cases}$$

Where w and b are parameters of the model. The solution of optimization problem is given by Lagrangian :

$$L_p = \frac{1}{2} \|w\|^2 - \sum_{i=1}^n \alpha_i [y_i(wx_i + b) - 1] \quad (1)$$

Where α_i are called the Lagrange multiplier. We can simplify the problem given by equation 1 as follows :

$$L_D = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{ij} \alpha_i \alpha_j y_i y_j x_i x_j \quad (2)$$

In several cases, linear solutions could not solve the optimization problem. In this situation, a non linear separator is required. The formulation of the problem is given bellow :

$$f(x) = \begin{cases} f(x) = wx_i + b \geq (1 - \xi_i) & \text{if } y_i = 1 \\ wx_i + b \leq (1 - \xi_i) & \text{if } y_i = -1 \\ \xi_i > 0, \forall i \end{cases}$$

The objective function will change as follows :

$$f(w) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n (\xi_i)^{\kappa} \quad (3)$$

Where C and ξ_i are specified by the user and represent the penalty of mis-classification. The Lagrangian is written as follows :

$$L_p = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n (\xi_i) - \sum_{i=1}^n \alpha_i [y_i(wx_i + b) - 1 + \xi_i] - \sum_{i=1}^n \mu_i \xi_i \quad (4)$$

We can however, simplify the problem given by equation 4 as follows :

$$L_D = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{ij} \alpha_i \alpha_j y_i y_j x_i x_j \quad (5)$$

The problem given by equation 5 becomes identical to the linear discrimination problem given by equation 2.

B. Multi-class classification

In order to treat multi-class problems by constructing binary problems, several methods have been proposed. There are three methods developed to deal with multi-class problems using SVM classifier at each node :

1) *One-against-one method*: To resolve multi-class problem, one-against-one method requires one classifier SVM_{ij} for each pair of classes (i, j) . It builds $[n(n-1)/2]$ classifiers for n -class classification problem. During the test phase, the test set is evaluated by all SVM_{ij} .

Let $E = (x_i, y_i)_{i=1, n}$, be a training data set, where $x_i \in R^n$ and $y_i \in \{1, 2, \dots, k\}$. For k class problem, the optimization problem to construct SVM_{ij} that separate two classes C_i and C_j is given as follows :

$$(P) = \begin{cases} \min_{w^i, b^i, \xi^i} \frac{1}{2} (w^{ij})^T w^{ij} + C \sum_t \xi_t^{ij} \\ (w^{ij})^T \phi(x_t) + b^{ij} \geq 1 - \xi_t^{ij} : y_j = 1 \\ (w^{ij})^T \phi(x_t) + b^{ij} \leq -1 + \xi_t^{ij} : y_j \neq 1 \\ \xi_t^{ij} \geq 0 : j = 1, \dots, k. \end{cases} \quad (6)$$

To determine the decision function ($f_{ij}(x) = Sgn(w_{ij}x + b_{ij})$) which separates classes C_i and C_j , we use Max-Win strategy :

$$Sgn(x) = \begin{cases} +1 : x > 0 \\ -1 : x \leq 0 \end{cases}$$

$$x \in \begin{cases} C_i : f_{ij}(x) = 1 \\ C_j : f_{ij}(x) = -1 \end{cases}$$

The process of Max-Win strategy is given as follows :

- For each x_i :

$$f_{ij}(x) = \sum_{j \neq i, j=1}^k Sgn(f_{ij}(x)) \quad (7)$$

- The class of x_i is obtained by :

$$\arg \max_{i:1, \dots, k} f_{ij}(x) \quad (8)$$

2) *One-against-all method*: The one-against-all method is simple and efficient. It requires n classifiers $SVM_i : (i = 1, n)$, for n -class classification problem. During the test phase, the test set is evaluated by the SVM_i . SVM_i which shows highest decision value is chosen.

Let $E = \{(x_1, y_{1j}), (x_2, y_{2j}), \dots, (x_l, y_{lj})\}$ be a training data set, where $x_{i(i=1,l)}$ represents the i^{th} observation and $y_{ij(j=1,k)}$ represents the j^{th} class of the i^{th} observation. For k class problem, the formulation of the j^{th} SVM is given as follows :

$$(P) = \begin{cases} \min_{w^j, b^j, \xi^j} \frac{1}{2} (w^j)^T w^j + C \sum_{j=1}^l \xi^j \\ (w^j)^T \phi(x_i) + b^j \geq 1 - \xi^j : y_j = j \\ (w^j)^T \phi(x_i) + b^j \leq -1 + \xi^j : y_j \neq j \\ \xi^j \geq 0 : i = 1, l; j = 1, k \end{cases} \quad (9)$$

We solve the problem in (9) and obtain k decision functions :

$$(P) = \begin{cases} (w^1)^T \phi(x_i) + b^1, \\ \vdots \\ (w^k)^T \phi(x_i) + b^k \end{cases} \quad (10)$$

The class of x_i is obtained as follows :

$$Class(x) = \arg \max_{(i=1, \dots, l)} ((w^j)^T \phi(x_i) + b^j). \quad (11)$$

3) *Directed Acyclic Graph SVM (DAGSVM)*: The DAGSVM method constructs also $[n(n - 1)/2]$ classifiers SVM_{ij} . During the test phase, it creates a list of all candidates classes. At each test, the class that obtained negative score is eliminated from the list.

IV. SVM FUZZY HIERARCHICAL CLASSIFICATION METHOD

The new method we propose in this paper supplies an alternative to the three methods : One-against-one, One-against-all and DAGSVM. Our method is based on a fuzzy hierarchical classification technique we developed for the specification software reuse [15]. It provides also advantages to treat hierarchical multi-class problems. The method we propose consists of three steps : (A) Training data set compression by K-Mean (B) Fuzzy hierarchical classification building and (C) Introducing membership function for training SVM.

A. Training data set compression

Several works focused on reducing the number of training data set of SVM [16]. The first step in our method is compressing training data set of SVM. We apply basic K-Mean algorithm in order to regroup similar data in the same cluster and reduce time spent in training data set of

SVM. The goal of this step is expressed by an objective function that depends on the proximities of the points to their centroids. To assign each object to the closest centroid, we apply equation 12 :

$$g_i = \frac{1}{m_i} \sum_{x \in C_i} x \quad (12)$$

Where g_i represents the centroid of cluster C_i , m_i represents the number of objects in the i^{th} cluster and x is an observation.

In order to measure the quality of clustering, we use the sum of the squared error (SSE), given by :

$$SSE = \sum_{i=1}^k \sum_{x \in C_i} dist(g_i, x)^2 \quad (13)$$

Where k represents the number of clusters.

B. Fuzzy hierarchical classification building

1) *Similarity measure*: The notion of a distance between x and y has long been used in many contexts as a measure of similarity or dissimilarity between a set's elements. In this work, we define a relative generalized Hamming distance δ to compute similarity between clusters which is defined by :

$$\delta(\varsigma_i, \varsigma_j) = \frac{1}{n} \times d(\varsigma_i, \varsigma_j) = \frac{1}{n} \sum_{i=1}^n |\mu_{\varsigma_i}(x_i) - \mu_{\varsigma_j}(x_i)| \quad (14)$$

Where n represents the number of clusters and $d(\varsigma_i, \varsigma_j)$ is the Hamming distance between clusters ς_i and ς_j .

Since $\mu_{\varsigma_i}(x_i)$ and $\mu_{\varsigma_j}(x_i) \in [0,1]$, $\forall i = 1, n \Rightarrow$

$$0 \leq \delta(\varsigma_i, \varsigma_j) \leq 1. \quad (15)$$

2) *Fuzzy subsets*: Let K be a universe of discourse, $A \subset K$, and $K = \{x_i\}$. An element x of K belonging to A is defined as : $x \in A$. Let $\mu_A(x)$ be a characteristic function whose value indicates whether x belongs to A according to :

$$\mu_A(x) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{if } x \notin A \end{cases} \quad (16)$$

The characteristic function $\mu_A(x)$ takes its values in the interval $[0,1]$. It is defined as a mapping :

$$\mu_A(x) : A \rightarrow \{0, 1\} \quad (17)$$

The fuzzy logic is based on partial membership function. An object is belonging to one or more than a class in the same case. Let A be a sub set, defined by its membership function μ_A . The membership function $\mu_A(x)$ of an object x used in fuzzy set theory is defined as follows : An object x does not belong to class C if the membership function $\mu_C(x) = 0$, belongs a little to class C if $\mu_C(x)$ border to 0, belongs enough to class C if $\mu_C(x)$ does not border to 0 nor to 1, belongs strongly to class C if $\mu_C(x)$ border to 1 and belongs completely to class C if $\mu_C(x) = 1$.

3) **Fuzzy operators:** Let A and B be fuzzy subsets of universe K . The fuzzy operators on the fuzzy subset A and B of K are given as follows :

- **Intersection operator (AND)**

The membership function used by [17] to define the set $(A \cap B)$, is given by the minimum of membership functions μ_A and μ_B as follows :

$$\forall x \in X : \mu_{A \cap B}(x) = \min\{\mu_A(x), \mu_B(x)\}. \quad (18)$$

- **Union operator (OR)**

The membership function defines the set $(A \cup B)$ is given by the maximum of membership functions μ_A and μ_B as follows :

$$\forall x \in X : \mu_{A \cup B}(x) = \max\{\mu_A(x), \mu_B(x)\}. \quad (19)$$

4) **Transitive closure of a fuzzy relation:** To extract ambiguous relationships between objects, we used the theory of fuzzy sets [17]. It is defined by their memberships function. In our work, we used Min-Max transitivity relation to find fuzzy relationships between objects :

$$\forall x, y, z \in K \times K \times K :$$

$$\mu_R(x, z) \leq \text{Min}_y[\text{Max}(\mu_R(x, y), \mu_R(y, z))] \quad (20)$$

We compute the transitive closure Min-Max given by equation 20 until we obtain transitive closure Γ equals to $\Gamma = R^{\kappa-1} = R^\kappa$ at κ levels. This equality assures the existence of a hierarchy. This relation gives the transitive distance Min-Max which locates the level of each objects and find the short link between these objects. Let $C_i = \{x_{i1}, x_{i2}, \dots, x_{in}\}$ and $C_j = \{x_{j1}, x_{j2}, \dots, x_{nj}\}$ be two clusters obtained by the similarity matrix. The fuzzy shortest link between two clusters is given as follows : $\Gamma_{ij} = \vee[(x_{i1} \wedge x_{j1}), (x_{i2} \wedge x_{j2}), \dots, (x_{in} \wedge x_{nj})]$.

C. Introducing membership function for training SVM

In this step, we train fuzzy SVM at each node of the hierarchy to subdivide the original problem into binary sub-problems.

Let M be a set of classes $C = \{c_1, c_2, \dots, c_k\}$, where k is the number of clusters obtained by K-Mean in the first step ($k \leq n$).

First, we compute the average transitive closure of all classes from the transitive closure matrix by the equation :

$$\bar{X} = \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n \Gamma_{ij} \quad (21)$$

Where n represents the number of values of Γ_{ij} in transitive closure matrix and Γ_{ij} represents fuzzy similarity value between C_i and C_j that are obtained by transitive closure.

Second, we compute the average of transitive closures of each class according to the following equation :

$$v_i = \frac{1}{k} \sum_{j=1}^n \Gamma_{ij}, j : 1, n \quad (22)$$

The fuzzy membership v_i , which is the average similarity between C_i and the rest ($k - 1$) of classes, is extracted from the transitive closure matrix.

Suppose that $E = \{(C_1, y_1, v_1), \dots, (C_k, y_k, v_k)\}$ a set of training data with associated membership, where $C_i \in R^k$, $y_i = \{-1, +1\}$ and $0 \leq v_i \leq 1$.

In our work and in order to handle multi-class with high precision, we introduced fuzzy membership function in the training SVM step. Each row i of the transitive closure matrix defines the membership between class i and the others classes. To construct positive and negative classes, we compute for each class C_i the membership value v_i . At each node of the hierarchy, the problem can be defined as follows :

$$SVM = \begin{cases} \{C_i\} \cup SVM_{ij}^+ : v_i > \bar{X} \\ \{C_i\} \cup SVM_{ij}^- : v_i \leq \bar{X} \end{cases} \quad (23)$$

The optimization problem given by our fuzzy SVM in (23) is given as follows :

$$\begin{cases} \frac{1}{2} w^T \cdot w + C \sum_{i=1}^k v_i \xi_i \\ y_i(w \cdot x_i + b) > 1 - v_i \xi_i \\ v_i \xi_i \geq 0 : i = 1, \dots, k \end{cases} \quad (24)$$

Where C, ξ_i represent the penalties of mis-classification and $v_i \xi_i$ represents error of classification with different weights.

Using the Lagrangian multiplier, the problem is given as follows :

$$\begin{cases} Max : w(\alpha) = \sum_{i=1}^k \alpha_i - \frac{1}{2} \sum_{i=1}^k \sum_{j=1}^k \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\ Subject : \sum_{i=1}^k y_i x_i = 0, 0 \leq \alpha_i \leq v_i C : i = 1, k \end{cases} \quad (25)$$

We repeat the process at each node of the hierarchy until reaching leaves containing only one class. Consequently, we obtain a descendant hierarchical classification represented by a succession of classes. Each class contains similar objects. The advantage of our method is that training data set of SVM obtained a priori by the transitive closure assures discriminating between positive and negative classes.

V. EXPERIMENTAL RESULTS

A. Data

In this paper, we compared the performance of our method with those of the methods : One-against-one, One-against-all and DAGSVM. We used three different problems available in [18]. The first problem is Iris database which contains 150 records grouped equally in three classes. The second problem is Glass database which contains 214 records distributed in six classes. The third problem is Letter database which contains 16000 records distributed in twenty six classes. We give detail of the three problems in Table I.

Table I
PROBLEM DETAIL

Problem	Data	Class	Attributes
Iris	150	3	4
Glass	214	6	9
Letter	16000	26	16

B. Experimental

- **Compression step**

To show how the compression step is usefull, we conducted two experiments. The first experiment consists in applying K-Mean to training data set step with the original data set replaced by clusters centroid. In the second experiment, we apply our method,without calling K-Mean. Table II shows results given by the two experiments.

Table II
COMPRESSION STEP INFLUENCE ON SVM-CHF PERFORMANCE

Data	With K-Mean			Without K-Mean		
	# SVM	Training time	Accuracy	# SVM	Training time	Accuracy
Iris	2	0.021	98.00	3	0.05	98.23
Glass	4	0.05	77.63	6	11	78.10
Lettre	21	110	98.35	24	255	98.45

In the first step, our method performs better in number of SVMs and training time criteria. Using K-Mean algorithm reduced automatically the number of SVMs and cost training time. In the second step we used the original data set which allows slightly better accuracy compared with accuracy result obtained in the first step. Since the two first criteria in the classification domain are very important, we introduced the K-Mean algorithm in the process of our method.

- **Kernel function**

In order to choose the best kernel function of each problem, we tested different kernel functions : Polynomial ($d=2,3,\dots,8$) and RBF ($\gamma = 0.1, 0.2, \dots, 1$). We choose only results where SVM performs well. The results are given in Table III.

For **Iris** ($k=3$) and **Glass** ($k=6$) problems, polynomial function gives best results. For **Letter** ($k=26$) problem, RBF function performs best. In our case, polynomial function

Table III
ACCURACY OBTAINED BY POLYNOMIAL AND RBF KERNEL FUNCTIONS

	Data							
	$SVM_{Poly:d}$				$SVM_{RBF:\gamma}$			
	2	4	6	8	0.1	0.2	0.4	1.0
Iris	0.98	0.95	0.94	0.94	0.97	0.96	0.90	0.96
Glass	0.66	0.77	0.76	0.69	0.66	0.69	0.72	0.65
Letter	0.54	0.67	0.88	0.87	0.78	0.93	0.98	0.92

performs better when the number of classes is small. High accuracy is obtained when $C = 2^{10}$, $C = 2^{11}$ and $C = 2^{11}$ for Iris, Glass and Letter problems respectively. The proposed method proved high performance for the three problems (Iris : 98.00%, Glass : 77.63% and Letter : 98.35%).

- **Accuracy comparison**

We use accuracy criterion to evaluate our results with results obtained by methods : One-against-one, One-against-all and DAGSVM. To obtain high accuracy, we tested our method with different values of C : $(2^2, \dots, 2^{12})$. Accuracy is obtained from confusion matrix. Our accuracy comparison results are compared with : One-against-one, One-against-all and DAGSVM (see Table IV). The proposed method proved high performance for the three problems.

Table IV
ACCURACY COMPARISON

Problem	One-against-one	One-against-rest	DAGSVM	Our Proposed Method
Iris	97.33	96.67	97.36	98.00
Glass	71.49	71.96	72.22	77.63
Letter	97.98	97.88	96.73	98.35

- **The fuzzy membership function influences on the classifier performance**

In this section, we tested the influence of the fuzzy membership function on the classifier performance. We varied v_i in the range from 0.1 to 0.8. Figure 1 shows that the high performance is obtained when v_i is equal to 0.31, 0.22 and 0.32 for problems Iris, Glass and Letter respectively. These fuzzy values are extracted from the transitive closure matrix. The values v_i are introduced to train SVM. Choosing v_i from transitive closure matrix allows our method to perform better.

VI. CONCLUSION

In this paper, we proposed a new fuzzy SVM hierarchical method to handle multi-class problems. The fuzzy hierarchical structure consists in subdividing the original problem into simple binary problems. Our method takes its advantage from using fuzzy hierarchical classification and fuzzy Support Vector Machine. Furthermore, it has the advantage of using only values from the similarity matrix

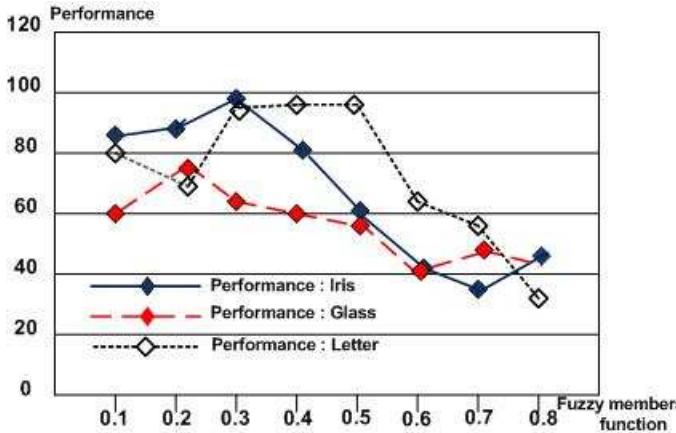


Figure 1. Fuzzy membership influence.

for the SVM training rather than using values randomly. Similarity matrix assures a priori separate classes in the hierarchy.

Unlike other classification methods, our method requires a number less than or equal to $(k-1)$ SVM classifiers from the root until leaves (see Table II). The number of SVM required reduce automatically the cost of training SVM time.

In this work, we find that introducing the membership values extracted from transitive closure matrix to SVM optimization problem gives a high accuracy.

Our future works consists in adapting our method to video sequencing problem in order to extract fuzzy relations between objects. Moreover, we will create a new dynamic kernel function to handle automatically classification process.

REFERENCES

- [1] S. Weiss and C. Apte and F. Damerau and D. Johnson and F. Oles and H. Goetz, *Maximizing text mining performance*. IEEE Intelligent Systems : 2–8, 1999.
- [2] X. Xie, *A validity measure for fuzzy clustering*. IEEE Transactions Pattern analysis and Machine Intelligence : 841–847, 1991.
- [3] D. Koller and M. Sahami, *Hierarchically classifying documents using very few words*. Proceedings of the 14th International Conference on Machine Learning, Nashville : 171–178, 1997.
- [4] T. Joachims, *Text Categorization with Support Vector Machines : Learning with many relevant features*. Proceedings of the 10th European conference on machine learning : 1998.
- [5] Y. Guermeur and A. Elisseeff and H. Paugam, *A new multiclass SVM based on a uniform convergence result*. IJCNN : 183–188, 2000.
- [6] C. Hsu and C. Lin, *A comparison of methods for multi-class support vector machines*. IEEE Trans Neural Networks : 415–425, 2002.
- [7] G. Zhang, *Support Vector Machine with Huffman tree architecture for multi-class classification*. Lecture Notes in computer Science : 24–33, 2005.
- [8] F. Schwenker, *Hierarchical Support Vector Machines for Multi-Class Pattern Recognition*. The 4th International Conference on knowledge-Based Intelligent Engineering Systems, Allied Technologies, Brighton, UK : 2000.
- [9] X. PENG and B. CHOI, *Automatic Web Page Classification in a Dynamic and Hierarchical Way*. IEEE International Conference on Data Mining : 386–393, 2002.
- [10] S. Dumais and H. Chen, *Hierarchical Classification of Web Content*. Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, Athens, Greece : 256–263, 2000.
- [11] L. Chun-Fu and W. Sheng-De, *Fuzzy Support Vector Machines*. IEEE Transactions On Neural Networks : 464–471, 2000.
- [12] T. Wang and H. Chiang. *Fuzzy support vector machine for multi-class text categorization*. Information Processing and Management : 914–929, 2007.
- [13] V. Vapnik, *Statistical Learning Theory*. John Wiley, Sons : 1998.
- [14] S. Tuffery, *Data Mining et statistique décisionnelle : L'intelligence des données*. Technip. Paris : 2007.
- [15] J. Nkoghe and K. Zeroual and W. Shengrui, *Specification Reuse : A Fuzzy Approach*. International Joint Conference on Artificial Intelligence : 1995.
- [16] B. Xiaojuan and S. Qilong and C. Hao and T. Xuyan, *Compression method based on training data set*. Journal of Systems Engineering and Electronics : 198–201, 2008.
- [17] L. Zadeh, *Fuzzy Sets*. Journal of Information and Control : 338–353, 1965.
- [18] C. L. Blake and C. J. Merz, *UCI repository of machine learning databases*. Available at <ftp://ftp.ics.uci.edu/pub/machine-learning-databases>, 1998.

MAXCLIQUE Problem Solved Using SQL

Jose Torres-Jimenez,
 Nelson Rangel-Valdez,
 Loreto Gonzalez-Hernandez
Information Technology Laboratory
CINVESTAV-Tamaulipas, Victoria Tamps., MEXICO
Email: jtf@cinvestav.mx,
nrangel@tamps.cinvestav.mx,
agonzalez@tamps.cinvestav.mx

Himer Avila-George
Departamento de Sistemas Informáticos y Computación (DSIC)
Universidad Politécnica de Valencia
Valencia, Spain
Email: hiavgeo@posgrado.upv.es

Abstract—This paper aims to show that SQL queries can be used to solve a well-known combinatorial optimization problem, the Maximum Clique Problem (MAXCLIQUE). This problem arises in many real world applications as computer vision and pattern recognition or coding theory to mention some of them. A clique of a graph is a complete subgraph, i.e., a graph where every pair of vertices is an edge. The MAXCLIQUE problem searches for the clique of the largest cardinality in a graph (the one with the greatest number of nodes). We propose a model based on SQL queries to solve this problem. We test our models in 62 random instances. Results show that the use of simple queries can yield solutions for the MAXCLIQUE problem in an easy and accessible form.

Keywords- Optimization; MAXCLIQUE; SQL query.

I. INTRODUCTION

Combinatorial optimization is a branch of applied mathematics and computer science that is used to solve problems like circuit design [1], scheduling [2], software testing [3], [4] and many other problems related with real world applications. A lot of problems presented in combinatorial optimization are best understood when they are abstracted in mathematical structures like graphs. Graph theory is the field of mathematics and computer sciences that studies all the aspects related with graphs. The importance of studying the combinatorial optimization problems is the wide application it has in real world problems.

One of the basic problems in graph theory is the MAXIMUM CLIQUE problem (MAXCLIQUE). This problem can be defined as the search of a complete subgraph of maximum cardinality, i.e. it contains the maximum number of vertices. Applications of this problem arises in coding theory [5], computer vision and pattern recognition [6], fault diagnosis [7] and protein structure similarity [8].

Several methods have been used to solve the MAXCLIQUE problem. In the literature can be found exact and non-exact approaches. A survey about this problem can be found in [9].

Most of the techniques used to solve the MAXCLIQUE problem rely in the use of a high level procedural language

like C [10], [11]. In this paper we propose two models to solve this problem using a well known non-procedural data access sublanguage, the Structured Query Language (SQL) [12]. SQL is easy to use and allows the users to express the desired results of a query in a high-level data sublanguage.

To the best of our knowledge, there is no reported approach that uses SQL queries to solve MAXCLIQUE instances. Moreover, we found no approach that solves any combinatorial optimization problem using SQL queries. The simplicity of the SQL language and the availability of database manager systems that allow the use of SQL motivate us to design a new approach that, based on SQL queries, could solve instances of the MAXCLIQUE problem. Basically, we present a query model which can be used to determine if a given graph has a clique of size k or smaller. The model was tested in a set of random generated instances.

The rest of the paper is organized as follows: section 2 presents the notation and formal definition of the MAXCLIQUE problem. Section 3 describes the optimization model based on SQL that solves this problem. Section 4 shows the experimental design used to test the proposed solution. Section 5 presents the conclusions derived from the results obtained when solving MAXCLIQUE instances through SQL queries.

II. MAXIMUM CLIQUE PROBLEM

In this section we give a formal definition for the MAXCLIQUE problem. After that, we show the solution of an instance of the MAXCLIQUE problem. Finally, we end the section analyzing the search space of the problem.

A. Formal Definition of the Maximum Clique Problem (MAXCLIQUE)

A graph $G = (V, E)$ is described by a set V of nodes and a set E of edges (or links between pair of nodes). The number of nodes of the set V represents the order of the graph G (denoted by $|G|$). A graph G is called *clique* when every pair of nodes $i, j \in V$, where $i \neq j$, is an edge

$(i, j) \in E$. The size of a clique is its order. From now on, a clique will be denoted by \mathcal{K} and its size by $|\mathcal{K}|$.

The MAXCLIQUE problem can be defined as follows: given a graph G , which subgraph $\mathcal{K}^* \subseteq G$ is the clique with the maximum possible cardinality $|\mathcal{K}^*|$? The set of subgraphs of G that answers this question is defined by the Equation 1. The subgraph \mathcal{K}^* is any of the subgraphs of G found in \mathcal{K}_{all}^* .

$$\mathcal{K}_{all}^* = \{\mathcal{K} : \mathcal{K} \subseteq G, |\mathcal{K}| = \max\{|\mathcal{K}| : \mathcal{K} \subseteq G\}\} \quad (1)$$

B. Example of a MAXCLIQUE instance

Following the definition already given, an instance of the MAXCLIQUE problem is a graph G . Figure 1 shows an instance of the MAXCLIQUE problem. This instance has 5 nodes and 5 edges.

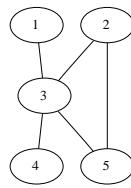


Figure 1. Instance of the MAXCLIQUE problem. The graph $G = (V, E)$ has the set $V = \{1, 2, 3, 4, 5\}$ and the set $E = \{(1, 3), (2, 3), (2, 5), (3, 4), (3, 5)\}$.

Figure 2 shows the SQL instance of the graph $G = (V, E)$ shown in Figure 1. The set of nodes V is represented in the table *nodes*; the field *v* is used to label the nodes of the graph and the field *s* to assign a weight to the nodes. The set of edges E is represented in the table *edges*; this table has two fields used to represent each edge of the graph as pair of nodes. In order to find clique graphs of size smaller than the predefined value k , a virtual node 0 and virtual edges between it and the original nodes are added to the graph instance; this will result in a participation of the node 0 in every clique found, but this node will not count in the size of the clique because its value in the field *s* is zero.

A SQL query can be used to solve the instance of MAXCLIQUE shown in Figure 2. First of all, this query involves a subset of size k . The query will try to find the greatest subset of size equal to or smaller than k that be a clique; it is possible because when there is no subset of size k the query looks by a subset of size $k - 1$ by including one time the zero node in the solution. Moreover, smaller subsets are possible by including as much zero nodes to the solutions as it is necessary.

The SQL query is shown in Table I. The *FROM* clause generates all the possible combinations of five copies of table *nodes*. The *WHERE* clause filters the combinations allowing only those in which a clique exists. The *ORDER BY* clause is used because the results of the query will give all the possible combinations of nodes that yields a clique.

<i>edges</i>		<i>nodes</i>	
<i>n₁</i>	<i>n₂</i>	<i>v</i>	<i>s</i>
0	0	0	0
0	1	1	1
0	2	2	1
0	3	3	1
0	4	4	1
0	5	5	1
1	3	3	1
2	3	4	1
2	5	5	1
3	4	5	1
3	5		
		(b)	
		(a)	

Figure 2. SQL tables encoding the instance of the MAXCLIQUE problem shown in Figure 1.

The evaluation of the existence of a clique is made by the clauses *EXISTS* which ask for the existence of all the edges required to form a clique. The descending order will give the maximum clique at the first entry of the results of the SQL query and the keywords *LIMIT 1* specify that we only want the first result (in case there is more than one solution). The name of the nodes that forms the clique and the size of the clique is specified in the *SELECT* clause.

Table I
SQL QUERY THAT SOLVES THE MAXCLIQUE PROBLEM INSTANCE SHOWN IN FIGURE 2.

```

SELECT
  V1.v, V2.v, V3.v, V4.v, V5.v,
  (V1.s + V2.s + V3.s + V4.s + V5.s)
FROM
  nodes as V1, nodes as V2, nodes as V3, nodes as V4, nodes as V5
WHERE
  EXISTS(SELECT * FROM edges as E WHERE (E.n1 = V1.v AND E.n2 = V2.v)
  AND EXISTS(SELECT * FROM edges as E WHERE (E.n1 = V1.v AND E.n2 = V3.v)
  AND EXISTS(SELECT * FROM edges as E WHERE (E.n1 = V1.v AND E.n2 = V4.v)
  AND EXISTS(SELECT * FROM edges as E WHERE (E.n1 = V1.v AND E.n2 = V5.v)
  AND EXISTS(SELECT * FROM edges as E WHERE (E.n1 = V2.v AND E.n2 = V3.v)
  AND EXISTS(SELECT * FROM edges as E WHERE (E.n1 = V2.v AND E.n2 = V4.v)
  AND EXISTS(SELECT * FROM edges as E WHERE (E.n1 = V2.v AND E.n2 = V5.v)
  AND EXISTS(SELECT * FROM edges as E WHERE (E.n1 = V3.v AND E.n2 = V4.v)
  AND EXISTS(SELECT * FROM edges as E WHERE (E.n1 = V3.v AND E.n2 = V5.v)
  AND EXISTS(SELECT * FROM edges as E WHERE (E.n1 = V4.v AND E.n2 = V5.v)
  ORDER BY 6 DESC LIMIT 1

```

The solution of the SQL query presented in Table I is shown in the Table II. The maximum clique size is 3. The columns represent the combination of the maximum clique and its size. Given that we search for a clique of size k , the number of columns in the solution will be $k + 1$, the first k columns are nodes that form the clique (a zero indicates a virtual node and must be ignored), the size of the clique is given in the last column.

Table II
SOLUTION OF THE MAXCLIQUE INSTANCE SHOWN IN FIGURE 2.

Clique						
V ₁ .v	V ₂ .v	V ₃ .v	V ₄ .v	V ₅ .v	size	
0	0	2	3	5	3	

C. Complexity Analysis of the MAXCLIQUE Problem

For a MAXCLIQUE instance, the search space in the SQL query is defined by the cartesian product performed in the clause *FROM*. Given that this product is between k tables of size $N + 1$ (the size of the clique and the number of nodes plus 1, respectively), the set of possible solutions for the problem has a cardinality of $(N + 1)^k$.

Several exact approaches have been proposed in the literature that solve the MAXCLIQUE problem [9]. These approaches generally work in the search space defined by all the subsets of size k or less of the set of nodes V of a graph $G = (V, E)$. This search space is equivalent to the cardinality of the power set of the set of nodes V , denoted by $|\mathcal{P}(V)|$ and defined in Equation 2.

$$|\mathcal{P}(V)| = \sum_{i=0}^N \binom{N}{i} \quad (2)$$

A comparison between the search space that potentially is searched by the SQL query is greater than the real search space. We present experimental evidence that the solution for smaller instances of the MAXCLIQUE problem are worth of attention through a SQL approach, taking into account that the high level of SQL avoids the programming of complex routines in low level languages.

Next section presents the generalization of the solution for the particular instance of the MAXCLIQUE problem already presented.

III. SQL APPROACH FOR SOLVING THE MAXCLIQUE PROBLEM

Given an instance of the MAXCLIQUE Problem as a graph $G = (V, E)$ and an integer k , $1 \leq k \leq N$, we propose an exact approach that finds a clique of size k or smaller in G . The approach consists on generating a query in the Standard Query Language (SQL). Once that the query has been created, it is executed in a database management system so that the solution is obtained.

The query shown in Table I presents the SQL query required to solve the particular MAXCLIQUE instance shown in Figure 2. Table III shows the generalization of that query such that any MAXCLIQUE instance defined according with the definition given at Section II-A can be solved using a SQL query and a database management. The query is given using the BNF notation [13]. The query uses the tables *edges* and *nodes*. The table *edges* contains the nodes of the graph V been solved; an extra virtual node called 0 is added to this set. A column associates the value 1 with each node in the original set V and the value 0 with the node 0. The table *edges* is the set of edges of the instance; this set includes extra virtual edges between node 0 and the rest of the nodes. The existence of edges between node 0 and the rest of the nodes allows this node to participate in any existing clique, but its contribution to the size of the clique is 0 so that

cliques formed by nodes of the original graph are preferred. In general, the inclusion of the node 0 and the edges with this node will enable the query to give as answer cliques of size smaller than k when a clique of size k does not exist. The solution of this query reports the nodes in the maximum clique found and the size of such clique.

Table III
BNF FORMAT OF THE GENERAL SQL QUERY THAT SOLVES THE MAXCLIQUE PROBLEM.

SELECT
<subset of nodes>,
<size of clique>
FROM
<subset definition>
WHERE
<constraint definition>
ORDER BY $<k + 1>$ DESC LIMIT 1
<subset of nodes> ::=
$V_1.v, \dots, V_K.v$
<size of clique> ::=
$(V_1.s + V_2.s + \dots + V_K.s)$
<subset definition> ::=
$nodes$ as V_1 , $nodes$ as V_2, \dots , $nodes$ as V_K
<constraint definition> ::=
$\text{EXISTS}(\text{SELECT } * \text{ FROM } edges \text{ WHERE}$
$edges.n1 = V_1.v \text{ AND } edges.n2 = V_2.v$
$\text{AND} \dots$

The structure of the query shown in Table III can be described as follows: given the graph $G = (V, E)$ as the SQL tables *edges*, *nodes* with fields n_1 and n_2 in *edges* and v, s in *nodes*, the *FROM* clause indicates the cartesian product of k tables as the nodes in the maximum clique of the instance that is going to be solved. The *SELECT* clause indicates the subset of maximum size that is a clique in the instance G . In addition, the *SELECT* clause include an extra field that represents the size of the clique found in the solution process. The *WHERE* clause will contain the condition that must be met so that a clique is formed by the SQL query; these instructions are logical ANDs of query's asking for the existence of every possible edge that must be contained in the clique, i.e., the query will ask for the existence of $\binom{k}{2}$ edges in this clause. Finally, the SQL query will sort the results by the extra field representing the size of the cliques. The descending order in combination with the clause *LIMIT 1* allow to identify a solution returned by the query; the descending order on the size of the cliques makes that the largest cliques appear as the first rows in the results, the clause *LIMIT 1* extracts only the first of them. Note that a convenient use of the clause *LIMIT 1* allows an extension in the results reported by the query, i.e. the query can report the c largest cliques found in the instance by specifying it in this clause (something like *LIMIT c*). In this way, if different cliques of the same size exists, they will be reported in the following tuples of the results.

In the next section is presented an experimental design to solve random instances of the MAXCLIQUE problem.

IV. EXPERIMENTAL DESIGN

In this section we present the experimental design done to test the model based on a SQL query to solve the MAX-CLIQUE problem. In order to test the performance of the query we solved 62 random instances of the MAXCLIQUE problem. The generators of the instances are described in the next subsection.

A. Random Instance Generators

Two models were considered for the generation of random instances of the MAXCLIQUE problem. The first model, or model *A*, is the well known Erdös-Rényi model [14]. This model works as follow: given the number of nodes n and a probability p , each edge $(i, j) \in E$ of the resulting graph $G = \{V, E\}$ is selected with a probability p , i.e., a random number is generated for each of the $\binom{n}{2}$ possible edges that a graph can have, and those edges with a random number smaller than p will belong to the graph.

Algorithm IV.1 shows the pseudocode of the model *A* for generation of random graphs. This algorithm takes as input the number of nodes n and the density p and gives as output a graph $G = (V, E)$, where $|V| = n$ and $|E| \approx p \cdot \binom{n}{2}$. Each edge $(i, j) \in E$ is selected with a probability p .

```
Algorithm IV.1: MODEL A( $n, p$ )
comment: Output : Graph G
for  $i \leftarrow 1$  to  $n$ 
    do  $\left\{ \begin{array}{l} \text{for } j \leftarrow 1 \text{ to } n \\ \text{do } \left\{ \begin{array}{l} q \leftarrow \text{RANDOMNUMBER}(0, 1) \\ \text{if } q \leq p \\ \text{then } \{\text{ADD}(G, } i, j) \end{array} \right. \end{array} \right.$ 
return ( $G$ )
```

A MAXCLIQUE instance using the algorithm previously described is shown in Figure 3. This instance has an number of nodes 6 and a value of p equal to 0.4.

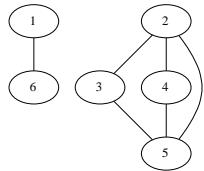


Figure 3. Random Instance of the MAXCLIQUE problem with 6 nodes and $p = 0.4$. The set of nodes is $V = \{1, 2, 3, 4, 5, 6\}$ and the set of edges is $E = \{(1, 6), (2, 3), (2, 4), (2, 5), (3, 5), (4, 5)\}$. A clique of size 3 is formed with the nodes 2, 3 and 5.

A disadvantage of using only the Erdös-Rényi model is that it doesn't model well the systems where the MAX-

CLIQUE problem finds its applications. While the Erdös-Rényi model is characterized by small clustering of the nodes with small average length paths, most of the systems that are found in the nature are best represented by highly clustered nodes with small average length paths (or as they are commonly referred, by small-world graphs). Due to this fact, we propose a second set of instances of the MAXCLIQUE problem to test the SQL query. The second model (or model *B*) is one of most widely used models for the generation of random small-world graphs, the Watts-Strogatz model [15].

The model *B* starts with a regular lattice and progressively rewrites the edges with a probability p . The rewiring process means that every edge $(i, j) \in E$ is disconnected with a probability p (the rewiring probability) from one of its end points and reconnected to another one. A regular lattice is a graph where each node has an edge with its z nearest neighbors, where z is called the coordination number. The pseudocode for the model *B* is shown in the Algorithm IV.2; this algorithm takes as input the number of nodes n , the coordination number z and the rewiring probability p and returns a random graph based on that values.

Algorithm IV.2: MODEL B(n, z, p)

```
comment: Output : Graph G = (V, E)
G  $\leftarrow$  BUILDREGULARLATTICE( $n, z$ )
for  $i \leftarrow 1$  to  $n$ 
    do  $\left\{ \begin{array}{l} \text{for } j \leftarrow 1 \text{ to } n \\ \text{do } \left\{ \begin{array}{l} \text{if } (i, j) \in E \\ \text{then } \{\text{REVERSE}(G, } i, j) \end{array} \right. \end{array} \right.$ 
return ( $G$ )
```

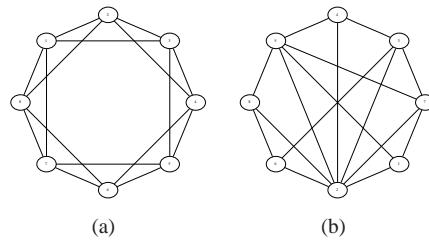


Figure 4. Random graph generated by model *B*: a) initial regular lattice with 8 nodes and coordination number $z = 2$; b) random graph generated after rewiring the edges with a probability $p = 0.50$.

Figure 4 shows an instance generated by the model *B*. Figure 4(a) presents the initial regular lattice of 8 nodes with coordination number $z = 2$. Figure 4(b) shows the resulting

graph after applying the generation model B with a rewiring probability of $p = 0.50$.

The next section presents the experimental results obtained from solving the random instances generated by the two models presented in this section. The instances were solved using, the SQL query described in Section III.

B. Computational Experiments

In this section we present the general overview of the experimentation, the instances solved by the SQL query and the features of the hardware and the database management system used in our experimentation. The results are summarized in several tables that contain information about the size of the clique found and the time spent by our approach to find it.

1) Features of the hardware and software: The random instances generator was implemented in C language and compiled with gcc. The instances were generated in a Desktop Computer with an Intel(R) Core(TM) 2 CPU 2400 6400 @ 2.13 Ghz processor, 3Gb of RAM and Ubuntu 8.10 Intrepid Ibex Operating System, and the query resulting from each instance was executed in MySQL 5.0.67.

2) Instances: The experiment was carried out using 62 instances which were created by the random instance generators described in the Section IV-A. Two sets of instances are included in the experiment. The first set (or set S_1) is shown in Table IV; this set is of small instances created using the model A and includes 14 instances with less than 30 nodes. The number of nodes and density of each instance is shown in columns 2 and 3, respectively.

The second set of instances S_2 includes larger instances created using the model B . Table V lists in columns 1 and 4 the different values for n, z, p used to create each of the 48 instances of the set. Basically, the number of nodes considered were $n = \{50, 70\}$, the coordination value z was varied from 5 to 15 and the rewiring probability values were $p = \{0.20, 0.30, 0.40, 0.50\}$.

3) Parameters of the test cases: The parameter required for the experiments was the size of the maximum clique (or k). The experiments were done in 2 different stages. In the first step the set S_1 was considered and tested with $k = 15$, the results from solving these cases are shown in the Table IV. The column 1 shows the instance. The column 2 shows the set of nodes that form the clique (a zero value represents a virtual node, for cliques of size smaller than k). The column 3 contains the size of the maximum clique. The column 4 shows the time consumed by the query.

According with the results shown in Table IV, the SQL query solved almost all the instances with less than 30 nodes and found cliques of size in the range from 3 to 15.

In the second step of the experimentation the set S_2 was considered. Also, in this experiment the value of the clique size was set to $k = 15$. The results from this experiment are shown in Table V. The first three columns shows the results

Table IV
RESULTS FROM SOLVING THE SET S_1 OF RANDOM MAXCLIQUE INSTANCES. THE CLIQUE SIZE WAS SET TO $k = 15$.

Cases	n	p	\mathcal{K}^*	ω	Time (sec.)
1	10	0.30	0 0 0 0 0 0 0 0 0 0 0 3 7 9	3	0.10
2	10	0.45	0 0 0 0 0 0 0 0 0 0 0 0 4 6 7	3	0.15
3	10	0.60	0 0 0 0 0 0 0 0 0 3 4 6 7 9	5	0.30
4	10	0.75	0 0 0 0 0 0 0 0 0 3 4 5 6 7 8	6	0.63
5	10	0.90	0 0 0 0 0 0 0 0 1 3 4 6 7 8 10	7	1.68
6	20	0.30	0 0 0 0 0 0 0 0 0 0 0 8 10 14 18	4	1.40
7	20	0.45	0 0 0 0 0 0 0 0 0 0 0 4 6 11 15 20	5	2.99
8	20	0.60	0 0 0 0 0 0 0 0 2 5 9 12 14 18 19	7	9.86
9	20	0.75	0 0 0 0 0 0 5 7 9 12 14 16 17 18 19	9	68.96
10	20	0.90	1 2 3 4 6 7 8 10 13 14 15 16 18 19 20	15	1222.83
11	30	0.30	0 0 0 0 0 0 0 0 0 0 1 2 14 22 25	5	6.28
12	30	0.45	0 0 0 0 0 0 0 0 1 12 14 20 22 25 28	7	19.10
13	30	0.60	0 0 0 0 0 1 6 8 14 17 22 23 25 27	9	86.22
14	30	0.75	0 0 0 2 5 8 14 15 17 20 22 23 27 28 30	12	815.30

for the instances with $n = 50$ nodes. The last three columns shows the results for the instances with $n = 70$ nodes. For each instance is listed de size of the maximum clique found $|\mathcal{K}^*|$ and the time in seconds spent by the query to find it.

Table V
RESULTS FROM SOLVING THE SET OF LARGE RANDOM MAXCLIQUE INSTANCES WITH THE SQL QUERY. THE VALUE OF THE MAXIMUM CLIQUE SIZE TO BE SEARCHED WAS SET TO $k = 5$.

Instance (n, z, p)	$ \mathcal{K}^* $	Time (sec.)	Instance (n, z, p)	$ \mathcal{K}^* $	Time (sec.)
(50, 5, 0.20)	6	7.80	(70, 10, 0.20)	8	335.51
(50, 5, 0.30)	6	6.24	(70, 10, 0.30)	9	209.38
(50, 5, 0.40)	4	5.36	(70, 10, 0.40)	6	151.12
(50, 5, 0.50)	4	5.04	(70, 10, 0.50)	6	126.88
(50, 6, 0.20)	6	10.73	(70, 11, 0.20)	9	504.95
(50, 6, 0.30)	5	8.68	(70, 11, 0.30)	7	235.76
(50, 6, 0.40)	5	8.10	(70, 11, 0.40)	7	165.93
(50, 6, 0.50)	5	8.30	(70, 11, 0.50)	7	168.17
(50, 7, 0.20)	7	18.72	(70, 12, 0.20)	9	637.30
(50, 7, 0.30)	7	14.56	(70, 12, 0.30)	9	377.82
(50, 7, 0.40)	6	14.02	(70, 12, 0.40)	7	280.25
(50, 7, 0.50)	5	12.20	(70, 12, 0.50)	7	229.70
(50, 8, 0.20)	8	31.33	(70, 13, 0.20)	10	819.89
(50, 8, 0.30)	7	18.99	(70, 13, 0.30)	9	569.04
(50, 8, 0.40)	6	16.90	(70, 13, 0.40)	8	428.36
(50, 8, 0.50)	5	15.30	(70, 13, 0.50)	7	331.47
(50, 9, 0.20)	8	35.06	(70, 14, 0.20)	10	1423.51
(50, 9, 0.30)	7	25.50	(70, 14, 0.30)	8	737.20
(50, 9, 0.40)	6	23.87	(70, 14, 0.40)	8	550.05
(50, 9, 0.50)	6	21.83	(70, 14, 0.50)	7	458.02
(50, 10, 0.20)	9	62.15	(70, 15, 0.20)	11	2081.27
(50, 10, 0.30)	7	38.39	(70, 15, 0.30)	8	719.54
(50, 10, 0.40)	7	38.07	(70, 15, 0.40)	8	617.33
(50, 10, 0.50)	6	32.21	(70, 15, 0.50)	7	512.06

The results shown in Table V show instances with an average small clique size (model A generated instances with greater cliques in graph with less nodes). The performance of the SQL query over the set S_2 is better than in the set S_1 , i.e. in some instances from the set S_1 the SQL query spent more time to find a clique of almost the same size than in the larger instances found in the set S_2 . A natural explanation of this behavior is that small-world graphs tend to be sparse, which weaken the possibility of finding large cliques, and the nodes are highly clustered, which affects the number of different subgraphs that could be a clique; these two characteristic can improves the performance of the SQL

query in the sense that the cartesian product exclude a large number of solution during the solution of the instance.

In general, the time spent by the SQL approach to solve the random instances varies from a few seconds to almost two hours. Note that this amount of time consumed by the query to find the maximum cliques is relatively small in comparison with the theoretical search space. For example, the instance 10 shown in Table IV have a clique of size $k = 15$ and the query only spent 1222.83 seconds to find it among the 31^{15} possible solutions. This performance can be mainly explained by the fact that the cartesian product is not done at once, instead it starts with two tables and continue adding them one by one until it is completed. Each time that two tables are combined, those tuples that do not match the conditions specified are left out from the rest of the operation; this action results in a considerable reduction in the search space that contributes to a quick localization of the wished result.

Finally, according with the results showed in this section, we can conclude that the model based on SQL queries can solve instances of the MAXCLIQUE problem when the number of nodes and/or the size of the clique searched are not too large.

V. CONCLUSIONS

This paper presents a novel approach for solving the MAXCLIQUE problem using a SQL query. The query was tested in a set of 62 random MAXCLIQUE instances created through the Erdős-Rényi and Watts-Strogatz models. The query performs well in small sparse instances, or instances where the maximum clique is small. The limitations of the model are given by the database management system used to solve the query.

The simplicity of the SQL approach makes it easier to use than procedural languages approaches, in the sense that it does not require complex structures nor programming skills to solve the problem. The performance of the SQL approach depends on the query optimization tools implemented in the database managements system. The results shown that it is possible to solve an important optimization problem using a high level non-procedural languages without coding a line.

Currently we are trying to extend the range in which a SQL query approach for the MAXCLIQUE problem works in reasonable time.

ACKNOWLEDGEMENTS

This research was partially funded by the following projects: CONACyT 58554-Cálculo de Covering Arrays, 51623-Fondo Mixto CONACyT y Gobierno del Estado de Tamaulipas.

REFERENCES

- [1] S. N. Bhatt and F. T. Leighton, "A framework for solving VLSI graph layout problems," *J. Comput. System Sci.*, vol. 28, no. 2, pp. 300 – 343, 1984.
- [2] Y. N. Sotskov, V. S. Tanaev, and F. Werner, "Scheduling problems and mixed graph colorings," *Optimization*, vol. 51, no. 3, pp. 597–624, 2002.
- [3] D. M. Cohen, S. R. Dalal, J. Parelius, and G. C. Patton, "The combinatorial design approach to automatic test generation," *IEEE Softw.*, vol. 13, no. 5, pp. 83–88, 1996.
- [4] C. C. Michael, G. E. McGraw, M. A. Schatz, and C. C. Walton, "Genetic algorithms for dynamic test data generation," in *Automated Software Engineering, 1997. Proceedings., 12th IEEE International Conference*. Washington, DC, USA: IEEE Computer Society, 1997, pp. 307–308.
- [5] V. Ustimenko and T. Shaska, "On some applications of graphs to cryptography and turbocoding," *Albanian J. Math.*, vol. 2, no. 3, pp. 249–255, 2008.
- [6] D. Conte, P. Foggia, C. Sansone, and M. Vento, "Graph matching applications in pattern recognition and image processing," in *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, 2003, pp. II–21–4 vol.3.
- [7] P. Berman and A. Pelc, "Distributed probabilistic fault diagnosis for multiprocessor systems," in *Fault-Tolerant Computing, 1990. FTCS-20. Digest of Papers., 20th International Symposium*, 1990, pp. 340 –346.
- [8] N. Malod-Dognin, R. Andonov, and N. Yanev, "Solving maximum clique problem for protein structure similarity," 2009. [Online]. Available: <http://www.citebase.org/abstract?id=oai:arXiv.org:0901.4833>
- [9] I. Bomze, M. Budinich, P. Pardalos, and M. Pelillo, "The Maximum Clique Problem," in *Handbook of Combinatorial Optimization*, D.-Z. Du and P. M. Pardalos, Eds. Kluwer Academic Publishers, 1999, vol. A, pp. 1–74.
- [10] G. Mulligan and D. G. Corneil, "Corrections to bierstone's algorithm for generating cliques," *J. ACM*, vol. 19, no. 2, pp. 244–247, 1972.
- [11] S. Tsukiyama, M. Ide, H. Ariyoshi, and I. Shirakawa, "A new algorithm for generating all the maximal independent sets," *SIAM J. Comput.*, vol. 6, no. 3, pp. 505–517, 1977.
- [12] C. Ordonez, "Programming the K-means clustering algorithm in SQL," in *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, NY, USA: ACM, 2004, pp. 823–828.
- [13] J. Friedman, "A computer system for transformational grammar," *Commun. ACM*, vol. 12, no. 6, pp. 341–348, 1969.
- [14] P. Erdős and A. Rényi, "On random graphs," *Publ. Math. Debrecen*, vol. 6, pp. 290–297, 1959.
- [15] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

Managing and Processing Office Documents in Oracle XML Database

Sabina Petride, Asha Tarachandani, Nipun Agarwal, Sam Idicula

Oracle Inc. USA

Redwood Shores CA, USA

{sabina.petride, asha.tarachandani, nipun.agarwal, sam.idicula}@oracle.com

Abstract - Office Open XML, an XML-based file format for office data, has been standardized, adopted by Microsoft Office 2007 and supported by other major office suites like OpenOffice. The question we try to answer in this paper is where Oracle XML Database (XMLDB) stands with respect to the new advances in XML Open standards. We present the XMLDB architecture that allows integration with Office Open XML. We discuss the implications for content search, generation and validation, brought by transparently storing office content in the XMLDB Repository. We explain how to use the XML storage model, XML indexes and XMLDB Repository features for improved querability, and how to integrate Office Open XML content search with relational, non-XML data sources available in a database.

Keywords - OOXML; Office 2010; XQuery; XMLIndex; Content Management Repository

I. INTRODUCTION

Office Open XML [1] (OOXML for now) has emerged as one of the industry standard file formats for representing word documents, spreadsheet, presentation and charts. It has been adopted by popular office applications: it is the file format for Word 2007. OpenOffice version 3.0 supports importing OOXML, with more products expected to follow.

With these emerging document standards come technical challenges. Systems are supposed to offer fast ingestion rates of data based on XML formats for data that has to be persisted on disks or filesystems, provide good querability and processing of such data, and integrate with easy to use and popular file content management applications. Moreover, users are expecting similar querability and accessibility options on their filesystem XML data as if it were stored in a database. Thus, more content management solutions have made the shift to (1) transparency with respect to the exact storage of the XML content, and (2) integration with popular document handling applications.

XMLDB has been around for almost a decade [2][3]. It allows for storing XML data in the database as a table column or in a filesystem-like Repository [4], that allows secure access to the data. Oracle XMLType is an abstraction and supports different storage models under the covers,

from object relational (shredded over relational tables and views), to a native binary XML format [5].

With respect to the storage transparency requirement (1), XMLDB already offers a filesystem like abstraction of XML content stored in the database, via the XMLDB Repository. Structured as well as unstructured content can reside in the repository and accessed via WebDAV or FTP protocol, as well as via PL/SQL APIs and SQL views. Furthermore, with the Oracle SecureFiles project [6], XML content can be stored in the server or a file system with relatively little performance difference. Here, we focus mainly on (2), and on detailing how to tune the Repository storage for best performance.

With large simplifications, both OOXML and Open Document Format (ODT for now) documents are ZIP-compatible archives that contain XML files together with files describing relationships among these; most notably, the actual content of the document is stored as XML. For simplicity, for the remainder of this paper, we will be talking about OOXML, with the note that the same approach can be taken for ODT and for that matter for any archive XML-based ZIP-compatible file format.

We present the architecture of a system that allows XML content manipulated in Office 2007 or OpenOffice to be transparently handled in the XMLDB repository and illustrate the key benefits of this system:

1) By transparently storing archived XML-based files in the XMLDB Repository, XML content can be navigated in a file-system fashion (via WebDAV).

2) As the XML content internally resides in the database, we maintain all the benefits of databases over filesystems: manageability, backup and recovery, security, integration with other features of the RDBMS.

3) New data conforming to the emerging open XML standards can coexist with data stored in the database; this allows for both XML content validation based on an RDBMS, as well as for dynamic content generation.

4) Internally, the system stores OOXML content in the binary-XML format allowing for good compression and disk space management, streaming XPath evaluation, piecewise updates, improved fragment-level querability, and integration with other database features like partitioning, utilities, native binary-XML midtier processing etc.

5) XMLIndexes are built on top of the binary-XML OOXML content; since query evaluation is internally optimized for binary-XML in the presence of XMLIndexes, this model gives efficient inter-document fragment-level search and intra-document XML processing.

6) Applications need not devise or implement their own authentication and authorization policy enforcement logic; instead, one can rely on the database authentication and the Access Control List (ACL) mechanism that protects XMLDB resources.

7) Straightforward integration of OOXML content with existing Oracle applications that render query output in formats chosen by the application. For instance, integration with BI Publisher for presenting fragment-level OOXML extraction results as PDF.

The paper is organized as follows: In Section II, we give the necessary background information for understanding the Oracle XML Database, with a focus on XML storage and indexing (Section II.A) and the Repository for XML resources (Section II.B). Section III details the system architecture for storing OOXML documents in XMLDB. Special considerations on dynamic content generation and content validation make the subject of Section IV, while Section V gives an evaluation of document and fragment-level security enforcement possible for OOXML data stored in the database. In Section VI, we discuss fragment-level query processing for OOXML, and Section VII details their usage for a project tracking Oracle database application. We conclude and point to related work in Section VIII.

II. BACKGROUND

We focus first on giving the necessary background information on XMLDB.

A. XML Storage and Indexing

XML content can be stored in the Oracle database either as large objects, in text format (CLOB), shredded as object-relational if schema-based (see [2]), or in the more recent binary-XML format (see [5]). With the binary format, XML tags are compacted into token identifiers. Besides reduced disk footprint, the binary-XML storage allows for fast query processing [5].

XML tables and columns can be indexed for improved XQuery performance. The XMLIndex [7] comes in a number of flavors: unstructured content can be fully indexed via an unstructured index, where essentially all paths in the XML content are indexed; semi-structured and structured content can be indexed via structured XML indexes, where the index creation statement specifies an XMLTABLE construct and the exact paths to be maintained by the index; finally, one can fine-tune an unstructured index by indicating that only certain types of paths be indexed via path-subsetting XML indexes. The application developer has the additional option of creating asynchronous XML indexes to defer index maintenance to a time when the server is less busy.

B. Repository Events

XMLDB provides an infrastructure for associating custom application code with XMLDB Repository actions. Various actions on the repository are defined as *events*;

examples of events are PRE-CREATE, POST-CREATE, PRE-UPDATE, POST-UPDATE, RENDER etc. Application code, called *event handlers*, is used to integrate application logic with the XMLDB repository. For example, a recycle-bin application can be built on top of XMLDB Repository using a PRE-DELETE event handler for all folders except the recycle-bin folder that creates a hard-link to the file that is to be deleted to recycle-bin.

Application specific event handlers are loaded into the Oracle Database and associated to all or certain resources in the repository via *resource configurations*, a particular type of resource. Once associated, the event listeners are used for any repository access – SQL views, PL/SQL APIs or protocols.

III. OPEN OFFICE XML DATA STORE AND RETRIEVAL - SYSTEM ARCHITECTURE

The XMLDB Repository is a filesystem-like abstraction that resides in the Oracle database and allows resources (with XML, text or binary content) to be stored and accessed either via protocols like WebDAV(RFC2518) and FTP, or via PL/SQL and JCR. Data in the repository is organized hierarchically, in folders and leaf resources, while internally it is stored in database tables.

MS Office 2007 uses WebDAV to save and open documents. The event handling mechanism of XMLDB Repository ensures that, when an OOXML document is saved under the specified path in the repository, the event handlers unzip it transparently (using the standard java.util.zip class) in the XMLDB Repository and the actual contents are moved to a Binary XML XMLType table. Similarly, all the component files are zipped on the way out of the repository at render time when the document is opened.

As the actual XML content is stored in an RDBMS, one can take advantage of the full-range of XML processing available in the database. The XML content table can be joined with relational tables present in the database. The event-based mechanism can be further exploited to dynamically build and enhance content that can be packaged to an application as OOXML, or to validate OOXML content against data available in a database. The system architecture is illustrated in Figure 1.

Figure 2 is an example of how the storage table and its index are created.

IV. CONTENT VALIDATION AND GENERATION

Two main applications of this framework are automatic content validation and generation.

A. Content validation

Storing the XML content of an OOXML in the database allows applications users to transparently validate the

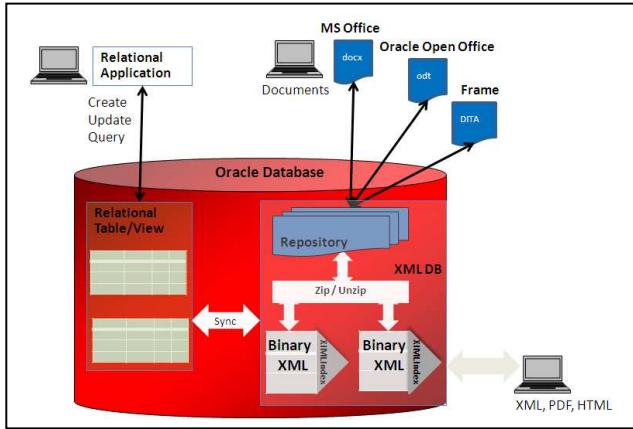


Figure 1: System architecture

OOXML content based on relevant data across multiple relational databases.

Consider for instance the case of a publisher database for managing the books submitted for review. Each book is a single Word 2007 document and, for the purpose of this section, we assume that the content is stored in a BOOK_XML table created via the statement shown in Figure 2. Author and book names, as well as the initial editing date and the actual publishing date are stored in a relational table of the form shown in Figure 3.

The system is supposed to validate the author and book names, as well as the date information in the Word document against the relational table. This can be easily incorporated into the application by issuing query checks involving the relational table BOOK_DATA and XML extract operators on the XML content in BOOK_XML. Figure 4 shows an example of a query against the XMLType table BOOK_XML that selects all the tags and their corresponding values from the document. Proper predicates with this query will achieve the desired results.

Figure 5 shows a simple query that finds the oldest publishing date of all authors who have at least one book in the category “Technical”. It involves both the XMLType table BOOK_XML and the relational table BOOK_DATA.

Using such joins, various validation rules can be applied automatically at ingestion time. For instance, to ensure that only authors of some technical books published prior to a fixed date are allowed to upload new books under a certain repository, the application event handlers can issue a query similar to Figure 5.

```
CREATE TABLE BOOK_DATA(
AUTHOR_NAME      VARCHAR2(4000),
BOOK_NAME        VARCHAR2(4000),
START_DATE       TIMESTAMP,
PUBLISHING_DATE TIMESTAMP);
```

Figure 3: Relational table

```
CREATE TABLE BOOK_XML OF XMLTYPE XMLTYPE STORE AS BINARY XML;

CREATE INDEX BOOK_XIDX ON BOOK_XML(OBJECT_VALUE)
INDEXTYPE IS XDB.XMLINDEX
PARAMETERS(' PATHS( INCLUDE(
/w:document/w:body/w:sdt//w:tbl)
NAMESPACE MAPPING
(xmlns:w="http://schemas.openxmlformats.org/wordprocessingxml/2006/main"))');
```

Figure 2: Binary XML table and XMLIndex

B. Dynamic Content Generation

Any content that can be retrieved from the database, can be added to an OOXML document. For instance, an application that stores book-related documents may have access to various relational databases for publishing companies extra information, or book prices offered by different vendors. Such additional content may or may not be stored as XML. Applications may expect to store a book document in the repository and, upon retrieval, to get back from the repository the book document together with the corresponding data from the other databases. Another desirable usage we have encountered comes from Excel applications: as loosely formatted Excel sheets are dropped in the repository, structured parts (e.g., columns that are titled “owner”, “user” or “manager”) are looked up against an LDAP database and edited to include a hyperlink with “mailto: <email address retrieved from LDAP database>”. This functionality is achieved by having a render event on the XML content resource issue queries on various tables,

TAG	VALUE
Title	The art of writing code
Category	Technical
Chapter	Chapter 1: Introduction
Chapter	Chapter 2: Understanding code
Section	Computer Languages: Similarity and Differences
Chapter	Chapter 3: Writing code

Figure 4: Selecting tags and corresponding values from Word 2007

```

SELECT FRAGVAL.VAL AS "Extracted Fragment"
FROM BOOK_XML BOOKS,
XMLTABLE( XMLNAMESPACE
  ('http://schemas.openxmlformats.org/wordprocessingml/2006/main' as "w"),
  '/w:document/w:body/w:sdt'
  PASSING BOOKS.OBJECT_VALUE
  COLUMNS TAG VARCHAR2(4000) PATH '/w:sdt/w:sdtPr/w:tag/@w:val',
  WHOLE XMLTYPE PATH '/w:sdt/w:sdtContent' ) TAGS,
XMLTABLE( XMLNAMESPACES
  ('http://schemas.openxmlformats.org/wordprocessingml/2006/main' as "w"),
  '/w:document/w:body/w:sdt'
  PASSING BOOKS.OBJECT_VALUE
  COLUMNS TAG VARCHAR2(4000) PATH '/w:sdt/w:sdtPr/w:tag/@w:val',
  VAL XMLTYPE PATH '/w:sdt/w:sdtContent//text()') FRAGVAL
WHERE TAGS.TAG = :search_in_tag AND
FRAGVAL.TAG=:return_tag AND
INSTR(UPPER(TAGS.WHOLE), UPPER(:search_string))>0;

```

Figure 5: Example of OOXML join with relational data

generate XML fragments from the queries results and update the XML content with them.

V. SECURITY AND PRIVACY CONSIDERATIONS

We mentioned in Section III that a user can open a *.docx* document in Word and save it in a folder in the XMLDB repository residing in the database. The user will need to provide valid database user/password credentials in order to connect to the repository. Once the user is authenticated, access control over OOXML data residing in the repository is handled, as for any other resources in the repository, via access control list (ACL) checks.

ACL checks are by default enforced at a document level. With XMLDB integration with Office 2007, certain fragments of the documents can be tagged with ACLs and honored by the application at the fragment level.

VI. FRAGMENT-LEVEL SEARCH AND RETRIEVAL

OOXML documents can be queried to retrieve

information just like any other XML data. This has a large number of applications. For example:

- 1) Searching using XQuery across a set of documents to retrieve relevant documents or parts.
- 2) Extracting out a specific part of the document such as abstract, instead of whole documents, to use for republishing, report generation etc.
- 3) Extracting out information from MS Word tables embedded in documents for application uses such as aggregation, population of relational tables etc.
- 4) Transparently control access to search results by taking advantage of the document and fragment-level security options when storing OOXML as content in the XMLDB repository.

When certain elements are tagged using content-controls, they can be queried in the WHERE clause as well as selected out as shown in Figure 6.

Note that unlike full document search, specific parts of the document can be searched like a table or tagged elements. For example, a repository of books can be searched with queries like *“Find all books and their authors that have at least one chapter with title containing keyword”*

```

SELECT BOOK.AUTHOR_NAME, min(BOOK.PUBLISHING_DATE)
FROM BOOK_DATA BOOK
WHERE BOOK.BOOK_NAME IN
(SELECT XMLCAST(XMLQUERY(
  declare namespace
  w="http://schemas.openxmlformats.org/wordprocessingml/2006/main";
/w:document/w:body/w:sdt[w:sdtPr/w:tag/@w:val="Title"]/w:sdtContent/w:t//text()
  PASSING BOOKS.OBJECT_VALUE
  RETURNING CONTENT) AS VARCHAR2(100) )
  FROM BOOK_XML BOOKS
  WHERE XMLEXISTS(
  declare namespace
  w="http://schemas.openxmlformats.org/wordprocessingml/2006/main";
/w:document/w:body/w:sdt[w:sdtPr/w:tag/@w:val="Category"] [w:sdtContent/w:t//text()="Technical"]
  PASSING BOOKS.OBJECT_VALUE))
GROUP BY BOOK.AUTHOR_NAME
ORDER BY BOOK.AUTHOR_NAME;

```

Figure 6: Querying content control

```

SELECT doc.c1.getStringVal() "LAYER NAME", doc.c2.getStringVal() "EFFECTS"
FROM BOOK_XML T,
XMLTABLE( 'XMLNAMESPACE
('http://schemas.openxmlformats.org/wordprocessingml/2006/main' as "w"),
for $i in $root//w:tbl[2]/w:tr
where fn:contains{$i}/w:tc[1], $layer)
return $i'
PASSING T.OBJECT_VALUE as "root", :p20_layer_name AS "layer"
COLUMNS C1 XMLTYPE PATH '/w:tr/w:tc[1]//text()', 
C2 XMLTYPE PATH '/w:tr/w:tc[2]//text()' ) doc;

```

Figure 7: Query the tables in a Word document for keywords

'haunted' and have been published in the last decade" and "Find all authors who have at least one book whose title contains keyword *'haunted'* and who have had at least five publications in the last 15 years". All this information, even though embedded deep inside the Office documents, can be retrieved.

Certain parts of the documents like tables and figures, can be searched without requiring any user input at all. For example, Figure 7 shows a query that looks for keyword defined by bind variable *:p20_layer_name* in the first column of the 2nd table of a *docx* document.

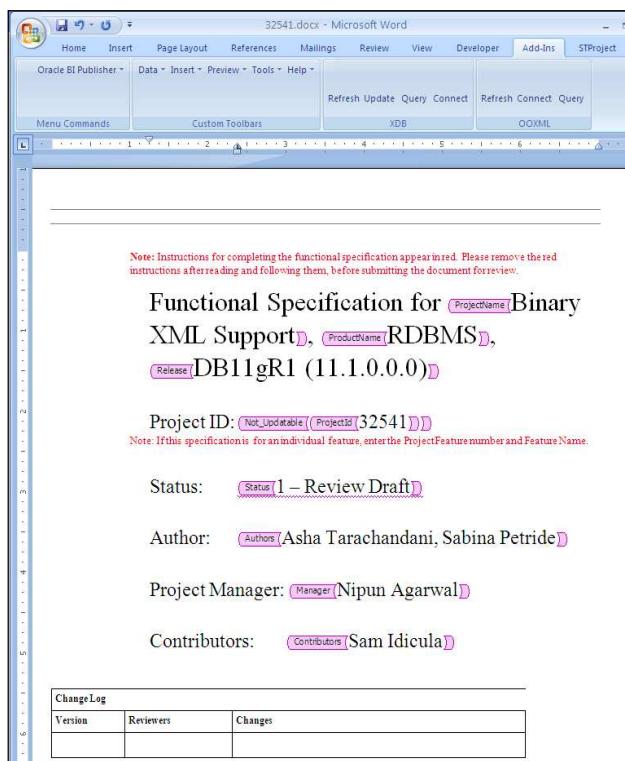
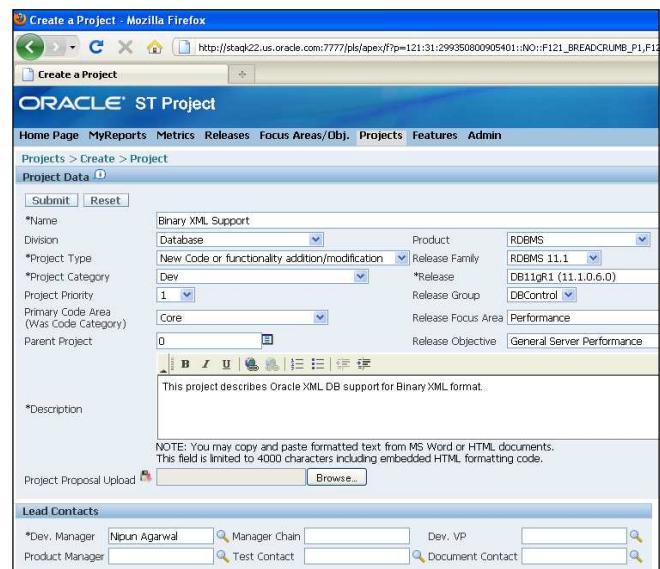
The search results can be returned in XML format and

integrated with various applications. For example, **Oracle BI Publisher** can be used to display the report in various formats such as PDF, Excel sheet etc. Similarly, the search results can be utilized to generate parts of other Office documents or they can be used to populate relational tables.

VII. CASE STUDY

We have built an internal application to track development projects in various releases of Oracle database. It allows online and real-time access to product development tracking tools. Part of the process involves maintaining a database of technical specification for products. Typical technical specifications are 2MB in size on average, with about 2000 projects with technical specifications per release (up to 4TB of content per release).

This site has an estimated 19K users. The typical searches and updates are real-time, while it is not rare for a DBA to

**Figure 8: Functional Specification Word document with custom tags****Figure 9: Creating a project transparently generates a Word document**

The screenshot shows a search interface for the XDB repository. The search query is "XML" in the "Description" section, and the result type is set to "Abstract". The search results table has two columns: "Document Path" and "Extracted Fragment". The first result is for document path "/xmlda/32475.docx", which contains an abstract about XML processing. The second result is for document path "/xmlda/32482.docx", which contains an abstract about NFS Version 4. The third result is for document path "/xmlda/32491.docx", which contains an abstract about XML DB limitations.

Document Path	Extracted Fragment
/xmlda/32475.docx	Prior to release 10gR2, XPath processing for schema-less documents was done by functional evaluation. XML DB added support for an XMLIndex in 10gR2 that allowed the ability to index both schema-less and schema-based documents stored as CLOBs. In 10gR2, the XMLIndex work focused on building a framework to improve the performance of XPath based queries for both fragment extraction and existence checking. XMLIndex can handle a large class of XPath expressions, including child axes, descendant axes, and equality and range predicates. It is also datatype aware with respect to value predicates by picking the datatype specified in the XMLSchema. This information is useful during implicit/explicit type coercions at the time of evaluation of an XPath. In 11gR1, we are enhancing the XMLIndex framework to improve performance, flexibility and meet more customer requirements. In this section, we present the motivation behind the various enhancements being introduced in 11gR1 and explain how some customers will be able to benefit from them.
/xmlda/32482.docx	NFS Version 4 [RFC 3530] is a distributed file system protocol that provides distributed transparent file access in a heterogeneous network. It is designed to be operating system and network transport independent. The NFS protocol definition is given as a set of well-defined procedures, with arguments and results, to access the underlying file system. It integrates features like file locking, strong security, operation coalescing, and delegation capabilities. NFS server running on XDB does not preclude other NFS servers running simultaneously on the same host.
/xmlda/32491.docx	Use this section to describe the project.

Figure 10: Search results – look for “XML” in “Description” section and return “Abstract”

perform offline batch updates.

By moving the project tracking database OpenOffice content to the XMLDB repository, the existing functionality is maintained, while new functionality like fragment-based search, publishing, 2-way sync with the relational table using triggers, is gained.

Figure 8 shows a typical functional specification. Word document where fixed fields, like title, author, project id etc. are tagged for easier search. In particular, note the *Not_Updateable* tag: the update event handlers associated with specification resources disallow changing XML content with this particular tag.

Figure 9 shows the web page for creating new projects that automatically generate *docx* file for the project. Figure 10 and Figure 11 show 2 web-based search interfaces – a standalone one and with BI publisher and result of a popular search. The search returns a document fragment matching the query, one for each specification document, as well as the Repository path of the specification, for easy full-document access.

VIII. CONCLUSION AND FUTURE WORK

We have presented the XMLDB solution for storing, querying and rendering Open XML content. Open XML content can be saved in XMLDB as a resource in the Repository providing direct WebDAV access to Office

The screenshot shows the Oracle BI Publisher interface. The search query is "Text: Vijay". The results table has two columns: "Document" and "Result". The results are as follows:

Document	Result
/xmlda/32458.docx	Vijay Medi
/xmlda/32460.docx	Vijay Medi
/xmlda/32506.docx	Asha Tarachandani, Vijay Medi
/xmlda/inPlaceEvolution_f2.docx	Vijay Medi
/xmlda/PullParser3.docx	Vijay Medi
/xmlda/saurabh.docx	Vijay Medi

Figure 11: Search results using BI Publisher

applications. For best performance, the XML content is stored in Binary XML format with a path-subsetted XMLIndex on it. The event-based mechanism is a powerful technique allowing dynamic content generation and validation, using *any* database data. Document and fragment-level access control enforcements methods available for XMLDB resources can be also applied.

Open XML integration with content management solutions for XML is carried out successfully also by MarkLogic [8] [9]. Their toolkits for Word, Excel and Powerpoint allow Open XML data to be saved in the MarkLogic server and subsequently queried via XQuery, manipulated and rendered. The main focus of the product is on search and retrieval of text and XML granular information. It allows for search results transformations, template-based content creation, and dynamic assembly of search results. There are a number of differences between ours and their approaches.

1) XMLDB Repository being part of the Oracle database, applications storing Open XML content in XMLDB implicitly benefit from all the general database features, like high availability, backup and recovery, security, utilities etc., as well as from more recent or particular features like smart lobes and secure files we can take advantage of when choosing binary storage for Open XML content.

2) Fine tuning of the actual storage and of the indexes on top of XML content is essential for good fragment-level querability. As detailed in Section II, the application developer storing Open XML in XMLDB has the option of specifying the XML storage format and of building XML indexes tailored to a specific set of queries or applications. To the best of our knowledge, there is no equivalent of path-subsetted XML index with MarkLogic, nor is the

application developer able to fine tune the storage and indexing method for particular query sets or applications.

3) XMLDB Repository events and resource configurations allow for custom and automatic work flow in applications. The application developer can use this single framework for quite different purposes, like dynamic content generation and content validation. Furthermore, this can also be used for 2-way synchronization between OOXML data and relational table with the help of event handlers and database triggers.

4) Both dynamic content generation and validation can use any data source in Oracle databases, which includes the entire XMLDB Repository. In particular, this covers non-XML, arbitrary relational data, while MarkLogic toolkits are tied to the XML content in their repositories. For the same reason, Open XML in the Oracle database is automatically available for manipulation to any database application.

Clearly, this is a functionality-only comparison. As products will become more mature and possibly other similar toolkits will be available, we expect benchmarks for Open XML and ODT handling in XML repositories to be set; we leave performance evaluations to future work.

REFERENCES

- [1] “Standard ECMA-376, Office Open XML File Formats”, 2006, <http://www.ecma-international.org/publications/standards/Ecma-376.htm>, 11.09.2010
- [2] Ravi Murthy, Zhen Hua Liu, Muralidhar Krishnaprasad, Sivasankaran Chandrasekar, et. al., “Towards an enterprise XML architecture”, Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 953–957, 2005.
- [3] Zhen Hua Liu and Ravi Murthy, “A Decade of XML Data Management: An Industrial Experience Report from Oracle”, IEEE 25th International Conference on Data Engineering, pp. 1351–1362, 2009.
- [4] Ravi Murthy and Eric Sedlar, “Flexible and efficient access control in Oracle”, Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data, pp. 973–980, 2007.
- [5] Ning Zhang, Nipun Agarwal, Sivasankaran Chandrasekar, Sam Idicula, Vijay Medi, Sabina Petride, and Balasubramanyam Sthanikam, “Binary XML Storage and Query Processing in Oracle 11g”, 35th International Conference on Very Large Databases (VLDB), volume 2, issue 2, pp. 1354–1365, 2009.
- [6] Niloy Mukherjee, Bharath Aleti, Amit Ganesh, Krishna Kunchithapadam, Scott Lynn, Sujatha Muthulingam, Kam Shergill, Shaoyu Wang and Wei Zhang, “Oracle Securefiles System”. Proceedings VLDB Endowment, volume 1, issue 2, pp. 1301–1312, 2008.
- [7] Geeta Arora, “XMLDB: Best Practices To Get Optimal Performance Out Of XML Queries”, Oracle White Paper, June 2010, <http://www.oracle.com/technetwork/database/features/xmldb/xmlquer-optimize11gr2-168036.pdf>, 11.09.2010
- [8] “Dynamic Enterprise Publishing: Accelerating Information Creation, Retrieval, and Delivery with Microsoft Office and Mark Logic”, MarkLogic White Paper, <http://www.marklogic.com/resources/dynamic-enterprise-publishing.html>, 11.09.2010
- [9] Mitchell Kramer, “BlueGuru JetBlues Content Management and Publishing System”, Case Study Prepared for Mark Logic by Patricia Seybold Group, 2009, <http://www.scribd.com/doc/17018347/MarkLogic-at-JetBlue-Cast-Study-Blue-Guru-CMS>, 11.09.2010

An Algorithm for Clustering XML Data Stream Using Sliding Window

Guojun Mao

College of Information

Central University of Finance and Economics

Beijing, China

maximmao@hotmail.com

Mingxia Gao, Wenji Yao

School of Computer Science

Beijing University of Technology

Beijing, China

gaomx@bjut.edu.cn;yaowenji@gmail.com

Abstract—This paper proposes an algorithm for clustering XML data stream using sliding window. It is a dynamic clustering algorithm based on XML structure. Firstly, we use level structure to represent XML document, which is based on temporal clustering feature. This structure is suitable for extracting information from XML document structure and calculating similarity between XML documents. Secondly, we use the sliding window technique, which adopts exponential histogram of XML cluster feature as a micro-cluster of it. By using the model, we can dynamically accept the new data and get rid of the old data thereby getting a better distribution feature of the current window. Finally, the experimental results based on real and synthetic XML datasets show that our algorithm not only achieves the real-time requirements of the online clustering, but also gains better clustering quality and faster processing speed.

Keywords-XML data stream; sliding window

I. INTRODUCTION

With the development of the Web applications, large amount of information is created, exchanged and stored in the form of Extensible Markup Language (XML) format. XML has the nature of flexibility and self-describing. Users can use XML documents to represent data according to their own needs. Many modern applications, such as stock information, real-time news subscription and release detection, often generate a new data format, which we call XML data stream. Discovering useful knowledge from XML data stream is an important research topic and faces many challenges. In order to discover more useful knowledge, many researchers focused on clustering XML documents and proposed a number of algorithms. However, the existing algorithms of clustering XML documents are mainly aimed for static datasets and generally need to repeatedly scan and parse the documents many times. They did not pay much attention to the time-varying online clustering context. The methods of clustering XML documents can be divided into two categories: (1) Researches based on semantics that consider both the structure of each node and its semantic information [13]. (2) Researches based on structure of XML that do not take the semantic information of XML documents into consideration [1].

In this paper, we propose an algorithm of clustering XML data stream called SW-XSCLS, which is based on the second strategy stated above. Our algorithm uses sliding window technology and takes exponential histogram of cluster feature as its summary of the data structure. It can

dynamically eliminate the outdated data and get a better understanding of the data distribution in the current window.

The contribution of our algorithm can be stated as follows: (1) we extend the conventional method of clustering XML documents to clustering the XML data stream; (2) we successfully apply sliding window technique to clustering XML data stream and create the SW-XSCLS algorithm; (3) the experimental results based on real and synthetic XML datasets show that our algorithm not only achieves the real-time requirements of online clustering, but also gains better clustering quality and faster processing speed.

The rest of this paper is organized as follows. Section 2 is about related work of processing XML documents and clustering data stream. Section 3 is the problem statement and Section 4 shows our method for clustering XML stream using sliding window. In Section 5, we show the experimental result and do some analysis from different aspect. Section 6 is about the conclusion and future work.

II. RELATED WORK

The traditional way of calculating the similarity between XML documents is the editing distance method. It computes the minimum cost for converting one tree to another tree. These operations include changing, inserting and deleting the node name. Costa et al. [1] use the idea of editing distance which converts each XML document to an ordered tree structure. They also use the “Jaccard” coefficient to calculate the similarity between XML documents. Wang et al. [2] define a standard for calculating the similarity distance. Their method need to analyze every XML document and use a directed graph to achieve the evaluation of similarity between XML documents. The accuracy of their method is not precise enough, because many XML documents that have different structures are possible to contain the same elements. So the standard has some limitations that may cause two different XML documents to be evaluated to have the same structure. Nayak [3] defines a level structure to represent the structure information of XML document and gives a similarity calculation criterion for this structure. But his method’s calculation result is related to the input order of the XML documents. Changing the input order will change the result.

There is also some work related to XML schema analysis. Vincent et al. [12] address the problem of extending the definition of functional dependencies in XML documents. Balmin et al. [9] exhibit an incremental validation algorithm for XML schemas and they show it is a significant

improvement over re-validation from scratch. Lu et al. [10] formalize the notion of the consistency of DTDs and propose a linear algorithm for checking the consistency. Elmasri et al. [11] introduce a design method for XML schemas based on well-understood conceptual modeling. They create XML schemas from a hierarchical point of view and generate SQL queries corresponding to the XML schemas.

In the clustering stream field, Zhang et al. [4] propose a method named BIRCH which incrementally and dynamically clusters incoming data points. Chang et al. [5] use two types of exponential histogram and sliding windows to cluster evolving data stream. Guha et al. [6] give constant-factor approximation algorithms for the k-Median problem and their method only need a single pass over the data. Zhou et al. [7] give a density based M-Kernel method for estimating data streams. Babcock et al. [8] present a novel technique for solving maintaining variance and k-median problems in sliding window model.

III. PROBLEM STATEMENT

A. XML Data Stream

In many researches XML data stream is defined as sequence of nodes obtained by the pre-order walk of the tree structure of XML. Processing XML data stream is analyzing these sequential nodes. But in this paper we define an XML data stream as follows:

Definition 1. An XML data stream is the sequence of XML documents ($X_1, X_2 \dots X_n$...) with timestamps. For any i ($i \geq 1$), X_i represents an XML document. The timestamp for each document is $T_1 \dots T_j$... and $T_i < T_j$, given $i < j$.

That is the arrival of each XML document is strictly time chronological. Since our study is based on the structural information, there is no restriction about the XML content.

B. Level Structure

This section mainly introduces the definition of the level structure of XML document and the process of analyzing XML data. The document is represented as an ordered tree with labels. Each tag (or element name) is denoted by a distinct integer according to their appearance order. By doing so, we eliminate the semantic information that can be inferred from the tag name and only concern about the structural information of the document. The level structure contains the hierarchy and context of the document. The multiple instances of values at same level are stored in an element since the occurrence number of an element is important for the clustering task.

The level structure of XML document contains the following information: the name, occurrence number and appear level of a node. Figure 1 shows an XML document and its level structure information.

Two level structures can be merged into a single structure. The principle is that merging the nodes at the same level. If multiple nodes with same name appear at the same level, we only save one of them. Figure 2 shows the merging process of two level structures.

An XML document can be represented by its level structure. The similarity between two XML documents is measured by the occurrence number of common elements in each corresponding level. Elements in different levels are assigned to different weights. The higher level has greater weight than the lower level. So documents with different root elements can be assigned to different clusters. Due to the page space limitation, the formula of calculating the similarity of two level structures can be referenced from Nayak [3].

C. Exponential histogram of cluster feature

Definition 2. Temporal Cluster Feature (TCF) is the collection of level structure of the XML documents in size n with the timestamp of $T_1, T_2, \dots T_n$. TCF is denoted as (LS, n, T) . The LS is the result after merging these level structures. The merging only occur between the same levels of two XML documents. The n is the number of level structure and T is the timestamp of the latest level structure in the temporal cluster feature.

The temporal cluster feature used in this paper is the extension of pseudo-time clustering features proposed in Chang et al [5].

Definition 3. Exponential Histogram of Cluster Feature (EHCF) is the collection of the TCFs. Given a set of documents $X_1 \dots X_n$ arrive at $T_1 \dots T_n$, and let $T_i < T_j$ when $i < j$. According to the order of arrival, these documents are divided into several groups denoted by $G_1, G_2 \dots$. Given $i < j$, all documents in group G_i arrive earlier than the documents in group G_j .

Exponential histogram is the first method to generate the summary information in the sliding window model which builds buckets according to the appearance order of the elements. The capacity of the bucket at different level increases exponentially in base 2. The number of buckets at the same level is no more than the number of barrels of a pre-defined threshold.

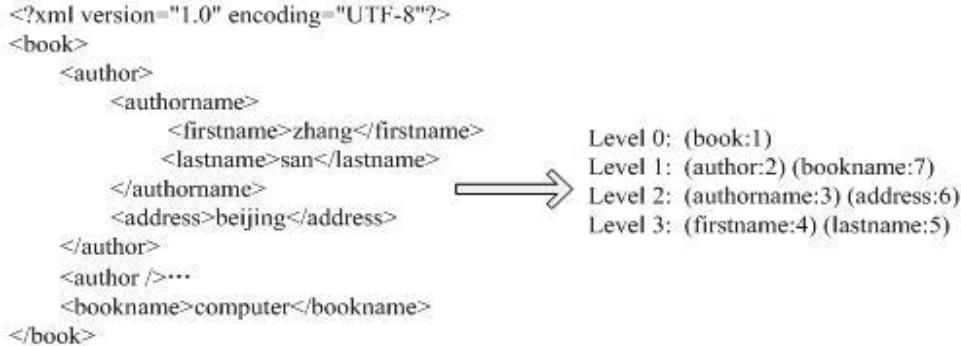


Figure 1. XML document and its level structure.

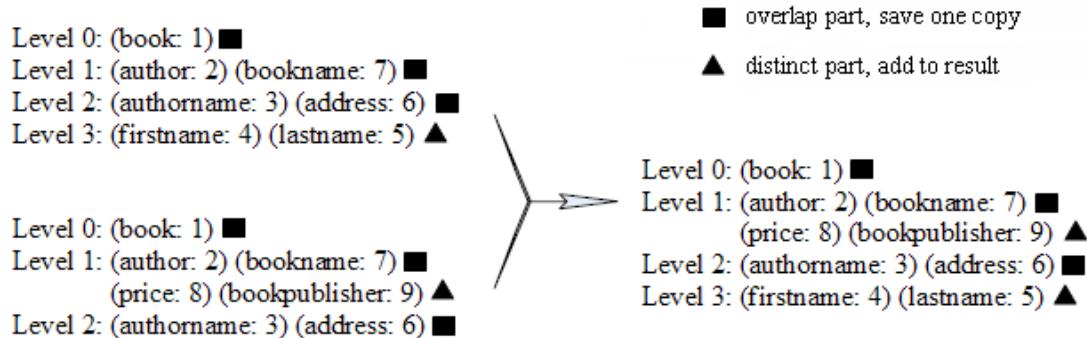


Figure 2. Merging two level structures.

IV. ALGORITHM FOR CLUSTERING XML DATA STREAM USING SLIDING WINDOW

This section focuses on the algorithm of clustering XML data stream using sliding window (SW-XSCLS). This algorithm can conduct clustering analysis for the XML data stream in the sliding window, which is maintained by a group of EHCF structures. This algorithm contains four parameters: DB is the XML data stream; W ($0 < W < 1$) is the similarity coefficient; N is the size of the sliding window; NC is the maximum number of EHCFs maintained by one window.

In this algorithm, the count value represents the number of EHCFs maintained in the current window. For each record x in the data stream, the first step is calculating the similarity (mostSimilaryValue) between x and the EHCF (h), which has the minimum distance with x . Then we test whether the mostSimilaryValue is greater than the similarity threshold W . If it is greater than W , then x is added to the current EHCF. Otherwise we will create a new EHCF, which only contains the element x and increase count value by 1. If the number of the EHCFs reaches NC, we need to merge the most recent two EHCFs.

The next step is updating the EHCF which contains the expire record. Since the operation is executed at arrival of each new record, so at any time there is at most one EHCF contains the expired elements. As we know that EHCF is the collection of TCFs and each TCF contains a timestamp T, we can find out the expired data by using TCF. If the timestamp T of the TCF is not belonged to N, we can delete the TCF. If

the last TCF in an EHCF is deleted, then the whole EHCF is deleted too. As a result, the count value, which is used to denote the number of EHCFs in the current window, will be decreased by 1.

We design two implementations for the calculating of the similarity between XML documents:

1. Merging all of the TCFs as a TCF in the EHCF (called allTCF) to calculate the similarity.
2. Using the latest TCF (called newestTCF) as the representative of the EHCF to calculate the similarity.

In both implementations, the first method merges all of the TCFs in the EHCF. So the accuracy of the calculating result is relatively high. The second method, which uses the latest TCF as a representative of the EHCF, has a faster calculating speed. In practice, using which kind of these two implementations is determined by the criteria that we concern about, that is the accuracy or the efficiency.

The whole process of the algorithm is described in Figure 3 as follows.

DB: XML data stream
W(0 < W < 1): similarity coefficient
N: window size
NC: the maximum EHCF number
GW-XSCLS (DB, W, NC, N)
begin
initial count to 0;
repeat each record x in DB
if x is the first record
generate an EHCF containing only TCF(x);
add EHCF to current window;
else
get h, the most similar EHCF with TCF(x);
get the mostSimilaryValue;
if mostSimilaryValue > W
Insert x into h;
else
generate an EHCF containing only TCF(x);
increase count by 1;
if count > NC
merge the two similar EHCF;
decrease count by 1;
end
end
end
if sum(x) > N
get EHCF, the EHCF containing the oldest TCF;
delete the Oldest TCF;
if EHCF is null
delete the EHCF;
decrease count by 1;
end
end
end

Figure 3. SW-XSCLS algorithm.

V. EXPERIMENTAL EVALUATION AND DISCUSSION

A. Datasets

We use both synthetic and real datasets in the experiments. The synthetic dataset is automatically generated by an XML generation tool called Oxygen. Because the number of real classes, the number of nodes and node levels of the artificial dataset can be controlled artificially, we can use it to test the processing time. The real dataset XMLFiles is the same as the dataset of static clustering algorithm XCLS in [3]. This dataset is composed of 460 XML documents from 23 natural areas, which includes 74 documents about films, 22 documents about universities, 208 documents about cars, 16 documents about literature, 38 documents about

companies, 24 documents about accommodation, 10 documents about tourism, 10 documents about orders, 4 documents about auction, 2 documents about stipulation, 15 documents about pages, 2 documents about books, 20 documents about games, 12 documents about associations, 2 documents about health care and 1 document about nutrition. The labels of documents range from 10 to 100 and levels from vary from 2 to 15.

B. Evaluation criteria

The performance of clustering XML documents is evaluated using the standard criteria named intra- and inter-cluster similarity. They are internal cluster quality evaluation criteria.

The intra-cluster similarity measures the cohesion within a cluster, how similar two documents in a cluster are. This is calculated by measuring the level similarity between each pair of documents in the cluster. The intra-cluster similarity of a cluster C_i [3] is the average of all pair-wise level similarities (between two trees) within the cluster, where n is the number of documents in C_i .

$$IntraSim(C_i) = \frac{\sum_{i=1}^n \sum_{j=i+1}^n LevelSim_{i,j}}{0.5 \times n \times (n-1)}. \quad (1)$$

The intra-cluster similarity of a clustering solution in the window $C = \{C_1, C_2, \dots, C_k\}$ [3] is the average of the intra-cluster similarities of all clusters taking into consideration the number of documents within each cluster, where n_i is the number of documents in C_i , N is the total number of documents and k is the number of clusters in the solution. The higher the intra-cluster similarity value is, the better the clustering solution is.

$$IntraSim = \frac{\sum_{i=1}^k IntraSim(C_i) \times n_i}{N}. \quad (2)$$

The inter-cluster similarity [3] measures the separation among different clusters. It is calculated by measuring the level similarity between two clusters. The inter-cluster similarity of the clustering solution is the average of all pair-wise level similarities of two clusters. The Level Similarity between two clusters is defined as similar to two documents, using the objects as clusters.

$$InterSim = \frac{\sum_{i=1}^k \sum_{j=i+1}^k LevelSim_{i,j}}{0.5 \times k \times (k-1)}. \quad (3)$$

In the experiment, unless specified, the parameters are set as follows: window size $N = 100$, similarity coefficient $W = 0.8$, the maximum number of histogram window $NC = 50$.

C. Experimental result and analysis

1) *Quality evaluation:* This section is the clustering quality comparison between our proposed algorithm (SW-

XSCLS) and the XCLS algorithm. Both algorithms use the real dataset. Since the clustering result of the algorithm XCLS is related to the order of the input, we execute the XCLS algorithm 5 times each with a different input order and calculate the average result as the evaluation criteria. Figure 4 shows the comparison of intra-cluster similarity. Figure 5 shows the comparison of inter-cluster similarity. From these figures we can see that the SW-XSCLS algorithm gets a better clustering result than the XCLS algorithm because the SW-XSCLS algorithm uses sliding window technology, which reduces the impact of outdated data in the result.

2) *Parameters*: Figure 6 shows the impact of execution time along with the similarity coefficient. The impact can be seen from the figure that with the increment of the similarity coefficient W , the processing time decreases. It is easy to understand that the larger of the W value is, the less number of EHCFs maintained in the window is. So the amount of calculation will reduce and the processing time will become less also. But this has a little effect. We can see from the figure that when the similarity coefficient increases from 0.6 to 0.9, while the processing time is increased by only 20 seconds. This is a small percentage of the total processing time. Figure 7 shows the window size and the impact on the execution time.

3) *Processing time*: We use XML documents with different levels and different number of nodes to evaluate the processing time for our algorithm. In order to observe the processing time, we introduce two concepts: the average number of levels and the average number of nodes. Because the synthetic dataset is available to get any number of

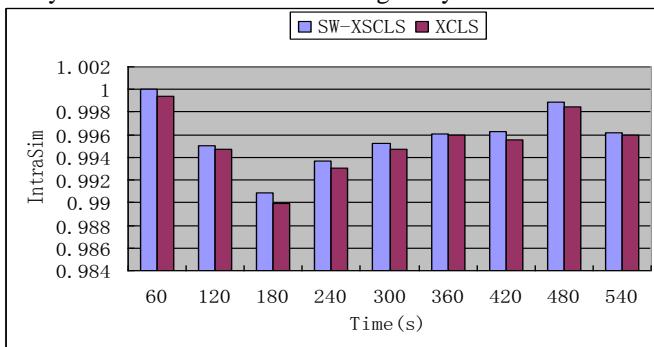


Figure 4. Intra-cluster similarity.

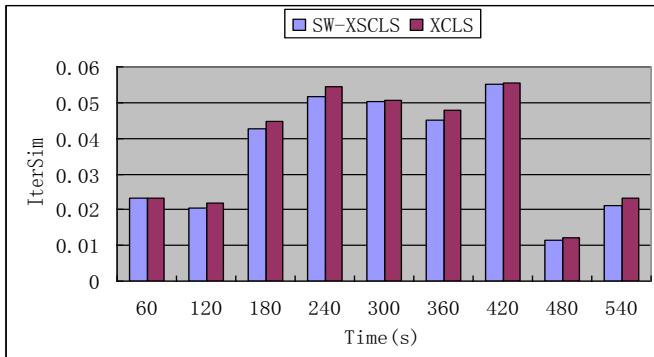


Figure 5. Inter-cluster similarity.

combinations of levels and nodes, so we generate a series of data sets. Figure 8 shows variation trend of the processing time of the algorithm tested on a series datasets which average number of levels changes and average number of nodes is fixed. Figure 9 shows variation trend of the processing time of the algorithm tested on a series datasets which average number of nodes changes and average number of levels is fixed.

VI. CONCLUSIONS

This paper proposed an algorithm for clustering XML data stream using sliding window. The algorithm is a dynamic algorithm based on level structure of XML documents. It can get a better clustering quality and a faster processing speed than traditional method. However, the existing clustering feature only takes into account the level structure information, ignores the semantic information of the data element. The effect is not good when processing XML documents with the same DTD or scheme. The future work will expand the existing expression on the type of data stream to meet the need for clustering data stream.

ACKNOWLEDGMENT

This research has been supported by Beijing Municipal Key Laboratory of Multimedia and Intelligent Software Technology and the National Science Foundation of China under Grants No.60873145.

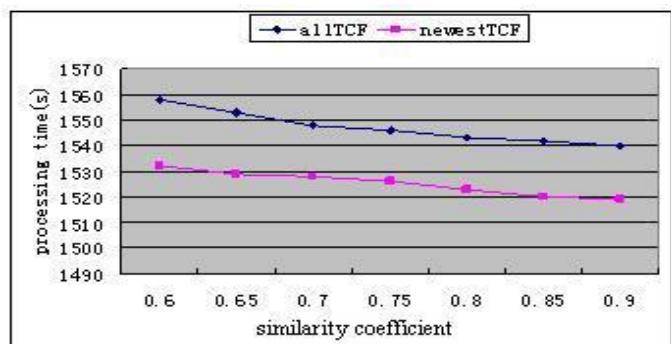


Figure 6. Changing the similarity coefficient.

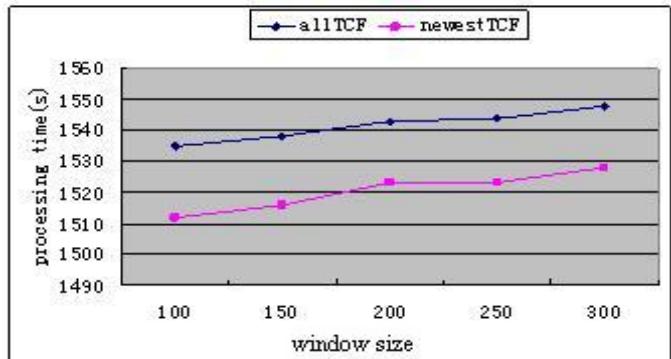


Figure 7. Changing the window size.

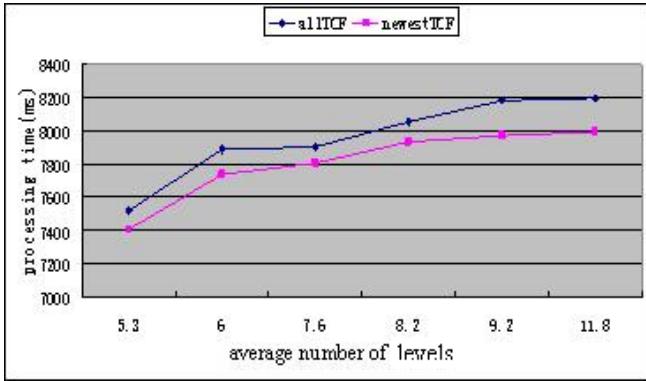


Figure 8. Processing time with number of levels.

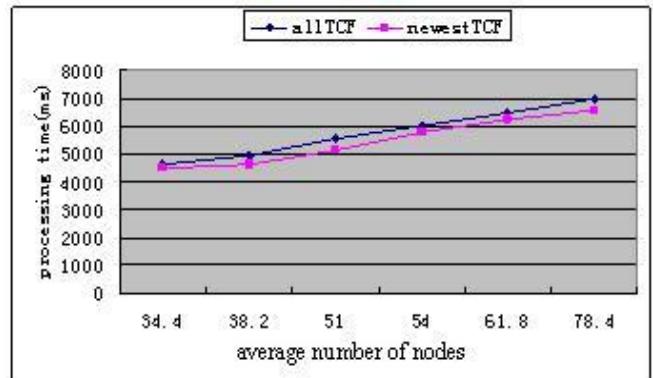


Figure 9. Processing time with number of nodes.

REFERENCES

- [1] G. Costa, G. Manco, R. Ortale, and A. Tagarelli, "A tree-based approach to clustering XML documents by structure," Proc. The 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, Springer-Verlag Press, Sept. 2004, pp. 137-148.
- [2] L. Wang, W. Cheung, N. Marnoulis, and S. Yiu, "An efficient and scalable algorithm for clustering XML documents by structure," IEEE Transactions on Knowledge and Data Engineering, vol. 16, Jan. 2004, pp. 82-96, doi:10.1109/TKDE.2004.1264824.
- [3] R. Nayak, "Fast and effective clustering of XML data using structural information," Knowledge and Information Systems, vol. 14, Feb. 2008, pp. 197-215, doi:10.1007/s10115-007-0080-8.
- [4] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: an efficient data clustering method for very large databases," Proc. ACM SIGMOD International Conference on Management of Data, ACM Press, vol. 25, Jun. 1996, pp. 103-114.
- [5] J. Chang, F. Cao, and A. Zhou, "Clustering evolving data streams over sliding windows," Journal of Software, vol. 18, Apr. 2007, pp. 905-918.
- [6] S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams," Proc. The 41st Annual Symposium on Foundations of Computer Science, IEEE Press, Nov. 2000, pp. 359-366.

- [7] A. Zhou, Z. Cai, L. Wei, and W. Qian, "M-kernel merging: towards density estimation over data streams," Proc. The 8th International Conference on Database Systems for Advanced Applications (DASFAA 03), IEEE Comput. Soc Press, Mar. 2003, pp. 285-292, doi:10.1109/DASFAA.2003.1192393.
- [8] B. Babcock, M. Datar, R. Motwani, and L. O'Callaghan, "Maintaining variance and k-medians over data stream windows," Proc. The 22nd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 03), ACM Press, Jun. 2003, pp. 234-243.
- [9] A. Balmin, Y. Papakonstantinou, and V. Vianu, "Incremental validation of XML documents," ACM Transactions on Database Systems, vol. 29, Dec. 2004, pp. 710-751, doi:10.1145/1042046.1042050.
- [10] S. Lu, Y. Sun, M. Atay, and F. Fotouhi, "On the consistency of XML DTDs," Data and Knowledge Engineering, vol. 52, Feb. 2005, pp. 231-247, doi:10.1016/j.datapk.2004.05.007.
- [11] R. Elmasri, Q. Li, J. Fu, Y. Wu, B. Hojabri, and S. Ande, "Conceptual modeling for customized XML schemas," Data and Knowledge Engineering, vol. 54, Jul. 2005, pp. 57-76, doi:10.1016/j.datapk.2004.10.003.
- [12] M. Vincent, J. Liu, and C. Liu, "Strong functional dependencies and their application to normal forms in XML," ACM Transactions on Database Systems, vol. 29, Sept. 2004, pp. 445-462, doi:10.1145/1016028.1016029.
- [13] Y. Guo, D. Chen, and J. Le, "Clustering XML documents by combining content and structure," International Symposium on Information Science and Engineering (ISISE 08), IEEE Press, vol. 1, Dec. 2008, pp. 583-587, doi:10.1109/ISISE.2008.31.

Societal View on Knowledge Representation and Management: A Case Study of an ICT Consulting Company

Cheng Chieh Huang

Dept. of Information Management
National Taiwan University,
Taipei, Taiwan
d94725007@ntu.edu.tw

Ching Cha Hsieh

Dept. of Information Management
National Taiwan University,
Taipei, Taiwan
cchsieh@im.ntu.edu.tw

Abstract—This article tries to understand knowledge management issues in markets instead of within organizations past literature focuses. It examines different knowledge integration and representation issues while facing different communities of practice in knowledge markets. In institutionalized markets, the knowledge commodity will tend to represent and integrate knowledge for dominant communities of practice in the regions or country. The practical issues, such as legitimacy of knowledge representation, complexity and dependences of knowledge boundaries, conflicts of social identities, bias of government technology policy are addressed. This paper contributes towards more societal views to understand knowledge exchange, representation value generation issues in knowledge markets.

Keywords-Knowledge Respresentation; Knowledge Market; Communities of Practices

I. INTRODUCTION

“Discussing logics for pushing ICT industry, and presenting to government project committee”, “Surveying the CIO’s IT spending practices and providing to the software and service vendors”, “Facing the media and various kinds of industries, speaking the ICT trends and visions” [11].

This is the work of a knowledge-intensive company which resides in the interfaces between organizations and social communities, bridges perceptual and practical differences among diverse communities in order to integrate and represent distributed knowledge. It produces and sells their knowledge in knowledge markets.

Most literature on knowledge management focuses on knowledge transfer, create, innovate issues within organizations, but neglect knowledge management issues in markets [1]. In fact, organizations absorb a lot of knowledge from markets, such as market reports, industrial news or consulting company advisors or other organizations. These knowledge sources, such as consulting company, how they collect and manage knowledge? How they represent knowledge? How they package their knowledge as commodity?

In this article, we discuss how the knowledge as commodity and mediate various communities in knowledge markets. Using community of practice concepts, we illustrate

knowledge integration and knowledge representation issues while facing multiple communities of practice in an ICT consulting firm. This paper seeks to contribute towards more societal views to understand knowledge exchange, value generation and management in knowledge markets.

In the following section, we first review literature of knowledge management and community of practice, and then propose an analysis framework. Second, we describe our methodology and contexts in our case. Third, the case story was illustrated using our research framework. Fourth, we present a discussion and fifth, we identify contributions, limitations and suggestions for future research.

II. LITERATURE REVIEW

The ‘community of practice’ has achieved prominence in the context of wider debates on knowledge, learning and innovation in organizations. Lave and Wenger [2] define the ‘community of practice’ as following:

An activity system about which participants share understandings concerning what they are doing and what means in their lives and for their community. Thus, they are united in both action and in the meaning that action has, both for themselves, and for the larger collective.

Brown and Duguid [3] also claims the knowledge shared and produced through the prism of practice, the way which work gets done. That knowledge is emergent and arise after the individuals begin to engage in collective practices. It focuses on practices, people rather than systems, technology that traditional knowledge management consideration.

Collective knowledge is not only embedded in communities of practice within organizations but also between organizations [3][4]. The members of communities of practice did not work side-by-side or meet face-to-face in everyday practices but create and share the professional knowledge through conferences, workshops, newsletter, web pages and the like. This is a kind of disciplinary, occupational or professional communities of practice; the knowledge is embedded in the networks, the broader structures [3][5].

III. RESEARCH FRAMEWORK

Based on the community of practice literature, we build up our framework (see Figure 1). Using this framework, we can examine knowledge representation and knowledge integration issues while facing multiple communities of practice in their different knowledge markets.

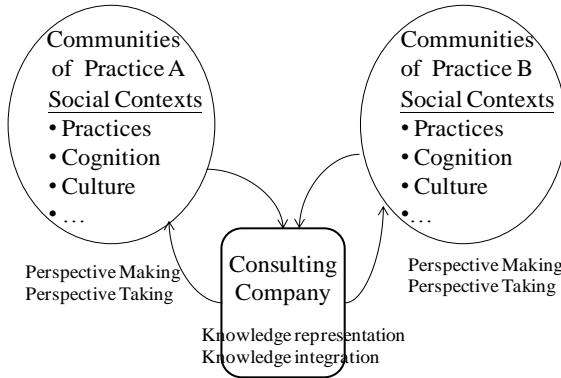


Figure 1. Research Framework

IV. METHODOLOGY

This researcher's fieldwork in the ITR institution comprised observing different members each day and working alongside many of them, interviewing the industry vendors, discussing the interview results, reviewing the survey results or presentation files, publishing reports, presenting in the seminars, responding to the customers, prospects or medias. In addition to the spontaneous, informal interviews that regularly occurred while the field researcher was observing the work. The interactions and dialogues among the participants were recorded in field notes, and the reflections of the practices also included [10].

After half-year observation, the researchers find interest topic about why the consultants of different practice teams generate knowledge differently. Semi-structured questions of formal interviews were addressed the consultants to ask the questions that allowed they provided their practices in acquiring, transferring, and generating knowledge. These questions include: How they interact with different communities of practice? How they generate new ideas or knowledge? How they inscribed the knowledge into their publications? All the interviews were tape recorded and translated to transcripts.

V. CASE STUDY

A. Case Contexts

The case company in this study is a well-known industry research and consulting firm, ITR institution. ITR is belonged to a legal body of financial group, which was established by Taiwan government more than 20 years ago, as a push for Taiwan ICT industry as well as an important think tank. Just because of the neutral role of the institution, and the importance of the Taiwan ICT industry in the global

ICT supply chain, ITR gets an irreplaceable status in Taiwan, and even all over the world. [6]

The institution divides the practice teams based on the product types, including such teams as PC, consumer electronic, network and communication, mobile communication, software and application, etc.

There are two kinds of main practices of ITR institution. One is the research reports they publish to their subscriber, called 'Product Practices'. Second are the consulting projects to customers, called 'Project Practices'.

No matter what kinds of work, the consultants collected data from specific industry, market data or other kinds of knowledge through face-to-face interviews, questionnaires or focus group methods. Then, they represent the knowledge, such as industry situations, market opportunities, trends, through market surveys or industry reports.

Their subscribers, customers or media read their publication and understand current and future market intelligences. The ITR institution or their consultants also try to enrich their influences in different communities of practice through their knowledge and publications.

B. Product Practices

Producing reports about the ICT trends and applications is main practices of ITR institution. Through understanding current situations of ICT industries, consumers or enterprises' intention to invest ICT, ITR consultants analyze the current situations and predict the future for customers in vary communities of practice.

Basically, ITR faces two industry communities: One is the well-known information and communication technology OEM (Original Equipment Manufacturing) industry. These vendors manufacture products for brand companies, such as HP, Dell and earn profits from their low manufacture costs (called 'OEM' community for short).

The other is information application and software industry (called 'Application' community for short) in Taiwan market. These companies sell their IT services or software products to enterprises in Taiwan.

While facing different communities, ITR practice teams (called 'OEM' practice team, 'Application' practice team) produce different reports, and encounter different knowledge management and representation issues.

1) 'OEM' Practice Team

'OEM' practice team faces major global information and communication technology OEM manufactures in the world. These manufactures produce products for global brand enterprises, such as HP, Dell and SONY. Most of these OEMs headquarters locate in Taiwan and compete orders from global brands. 'OEM' practice team's consultants analyze the shipment values and average selling prices by quarterly, and regularly provide reports to ITR institution's customers.

Noticeably, these companies manufacture major information and communication technology products in the world, such as NB, PC (i.e. 90% quantities produced by these companies). Global or domestic securities firms and

investment institutions are interested in purchasing these reports. Thus, the profit of ‘OEM’ practice team is stable. However, the institution also encounters intense competition from other consulting companies.

For ‘OEM’ industry community, ITR institution’s position is neutral, and it provides reliable statements for securities, investors or media. But, the shipment values, quantity or average selling prices in reports are not as important reference sources for ‘OEM’ community itself. A senior consultant who left ITR institution and worked for an OEM manufacturer said:

“For us, shipment quantities in reports are not important! We are more interested in understanding the movements of other manufacturers!”

However, since ‘OEM’ practice team’s quarterly reports on shipment values, shipment quantities of the manufacturers will influence stock market and economic in Taiwan and even represent global ICT boom. Also, it faces challenges from other competitors. Thus, consultants should be cautious about the numbers and trends they write in reports. In researcher’s participant observation, consulting managers of the practice team seriously examine the sources of each consultant’s numbers, trends and their reasons to shipment forecast. These reports should clearly define the logic of shipment forecast or trends. A consultant describes their logic of forecast:

“The logic of forecast is based on global market, product trends and movements of major global brand companies to analyze impacts on shipment quantity and average selling prices of OEM products.”

2) ‘Application’ Practice Team

‘Application’ practice team faces local software and IT services companies in Taiwan. These companies are not valued by shipment quantity like ‘OEM’ industry community, and they cannot compete in global market. Competition information in the community is not so important. For them, it is critical to understand their clients’ intention to buy their products or services. Consultants of practice team analyze market situations through market surveys or focus group methodology.

Since ‘Application’ community refers to small and medium firms, in comparison to manufactures in ‘OEM’ industry community, most of the firms in ‘Application’ industry community cannot afford to purchase these reports. Therefore, ‘Application’ practice team consultants tend to engage more government projects to earn more profits. Due to less time in product practice, consultants usually exploit reports from engaging in government projects. Thus, ITA’s clients usually complain reports do not meet their requirements. A business representative of ITR institution suggests that,

“We have been complaining about it. We expect the consultants to write more reports fulfill the clients’

requirement instead of exploiting reports from government projects engagement.”

However, for ‘Application’ practice teams’ consultants; it is also not easy to integrate knowledge related to various consumers or enterprises’ intention in different industries. Thus, the profits of these reports are low and also competitors are few.

ITR institution is neutral identity for ‘OEM’ industry community; however, it plays semi-official role for the firms in software and application community; the local small software and IT services firms expect to strive for some funds or influence government technology policy from ITR institution. A ‘Application’ practice teams’ consultants indicates,

“Most of our interviewees are the senior managers, such as general managers or CEOs. They are more familiar with their own industry than we young people! They are willing to spend time in talking to us since we represent the government. They would like to provide the suggestions of policy to government or understand funds opportunities from government through us!”

A CEO of a firm in software and application industry community suggests,

“I know that you are not the major decision makers (government policy), but I believe that I can try to convince each person in order to enhance the possibility to change government policy! “

Based on the above, the knowledge representation and knowledge integration issues are different when dealing with different communities in varies social environments.

C. Project Practice

Another practice of ITR institution is government projects engagement. ITR helps to understand the industry and market situations in order to propose effective projects to solve industrial issues and problems. Regarding the role of ITR institution in the government projects, the output reports allow government policy makers to understand industrial problems and also convince the reviewers (neutral scholars and experts) to agree the projects’ directions.

Since ITR institution is familiar with industry communities and knows how to represent the trends and gaps of industry, government policy makers or company want to get the government projects will very like to invite ITR institution in order to convince the reviewers to get projects. In the proposals, various companies which want to get the government projects must incorporate their original positions and demonstrate their logic of proposal to meet the industrial demand in order to persuade policy makers and reviewers. Thus, ITR institution tends to play the role as the main participant in knowledge integration and knowledge representation in the proposal.

However, there will be the logical conflicts in knowledge integration and knowledge representation. Should ITR institution integrate or represent the project partners' logic? Or ITR integrate and represent real ICT knowledge and represent the real industrial logics?

Noticeably, when industry trend is negative, government projects for pushing this industry will be meaningless. Once, a junior consultant published a report that indicated a product would not be future trend. The statement was widely used by the press. The consultant also involves in a project to promote the product, and partners of the government project called to complain about his opinions. A senior consultant advised the junior consultant:

"The contents of report should be presented tactfully and we should be more careful about the products we are involving in promoting!"

Besides partners, the thoughts of different government departments will influence ITR's representation of government project reports. Some governmental departments intend to promote information communication technology hardware manufacturer industry in Taiwan, some expect to enhance the industries with inferior global competitiveness, and some suggest enhancing information technique application in different industries. The most significant ability of ITR institution is to integrate various kinds of knowledge and represent the different logics or reasoning to reviewers or decision makers in the projects.

ICT trends in reports must fulfill interests of project members, but not always the real industry trends or companies' needs.

VI. DISCUSSION

A. Dependent Relationships and Knowledge Integration

In this study, the ITR consultants face different demands from different communities of practice and encounter different knowledge management and knowledge representation issues (see Figure 2 and Table 1).

For instance, 'OEM' practice team's consultants provide international and local media and investment institutions to understand current situations of the major global ICT OEMs. Demand of international, local media and investment institutions for knowledge is upon shipment quantities, shipment values or average selling prices from 'OEM' manufactures. For consultants, being trust in 'OEM' industry community in order to obtain the related information will rely on their experience and relationship maintained with their informants. We call the knowledge dependent relationship is 'partnership dependences'.

With costs consideration, 'OEM' practice team's consultants usually collect competition information from less than ten major 'OEM' firms and integrate their perspectives on industry trends. Reports from 'OEM' practice team always take angle from big vendor's perspective that becomes constraint devices [7] represent the perspective of large OEM firms and screen small firms' opinions.

In comparison to 'OEM' practice team, 'Application' teams' reports are integrated knowledge related to ICT spending intentions and applications of thousands of small and medium enterprises. ITR consultants acquire information by significant questionnaire surveys from enterprises' ICT spending intentions and sell to software and application vendors, we call the relationship is transactional dependences (see Figure 2).

ITR's reports for software and application community will not be constraint devices representing large-scale enterprises. However, without sufficient resources to acquire perspectives from thousands of companies, consultants cannot produce reports with specific industrial perspectives. Thus, software and application teams' reports are not valued by software and application community which tends to have many complaints that not in depth analysis for software and applications community demands.

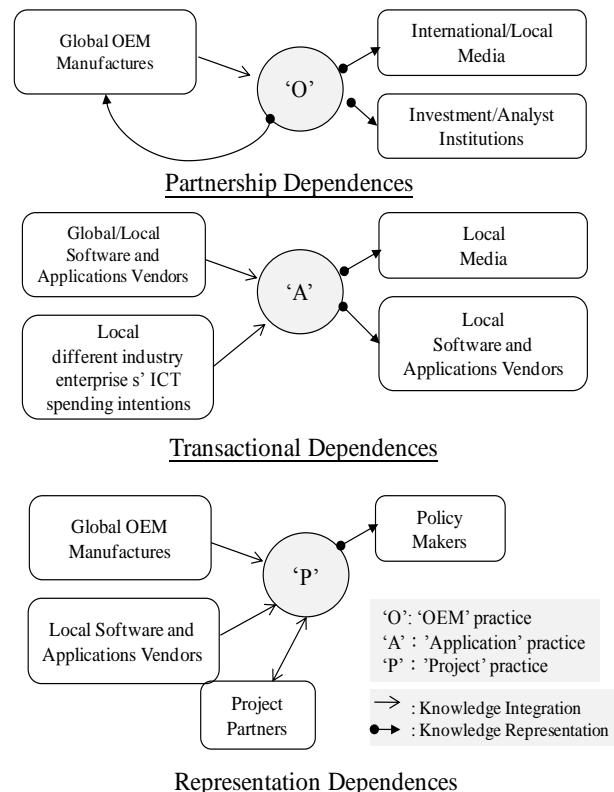


Figure 2. Three Different Dependent Relationships in ITR

The relationships between ITA consultants and industry community are representation dependences (see Figure 2). The ITA consultants represent industry community' issues to policy makers. The project practice encounter issues that to represent project partners' interests not industry community's.

Therefore, complexity and relationship dependences of communities will result in different problems of knowledge integration and representation (see Table 1).

TABLE I. ISSUES OF KNOWLEDGE INTEGRATION AND REPRESENTATION IN DIFFERENT PRACTICES

Practices	Knowledge Integration and Representation Issues		
	Demand	Integration Issues	Representation Issues
'OEM' practice team	shipment values or average selling prices	trust	only represent OEM logics
'Application' practice team	enterprise ICT spending intents in various industries	integrate various knowledge according to various industries	various reasoning and logics
'Project' practice	project partners' objectives	integrate different knowledge to fulfill partners' reasoning	logics for partners' reasoning, not the industry logics

B. Power and Knowledge Representation

This study also demonstrates that knowledge representations could become the resource of power.

Regarding ITR institution's creation of report in project practice in this study, ITR institution consultants integrate various kinds of knowledge and represent the logic that fulfill the partners' interests. For example, represent the logic that promotes some kind of partners' technology development benefit to industry, but in fact, industry has developed similar technology.

Knowledge representation becomes the resource to gain projects. The knowledge representations are not reflection of practical knowledge, but creation of 'reality' [9]. Thus, resources and funds of government subsidized to project partners, and they might not the real practical demands for industry. Reports produced by project practice are no longer to communicate between policy makers and industry communities.

Likewise, 'OEM' practice, demand from media and analytical institutions for shipment numbers leads to knowledge representations of large ICT hardware OEMs. With long-term concern, knowledge representations become the habitual way to represent. An 'Application' practice team's consultant criticized one report from 'OEM' practice team, "It is totally from the perspective of OEMs, and cannot probe into real industrial problems". It is the obstacle for 'OEM' practice team to create knowledge with new perspectives.

However, intentions and perspectives from different industry enterprises' ICT spending are various, and thus, knowledge representations are inconsistent. It is not easy for ITR consultants to represent specific industry logic and get their legitimacy in software and application industry community.

Thus, knowledge representations are influenced by power of communities, consistency of knowledge representations and institutionalization of long-term relationship. Knowledge

representation is not only a selection but also a deflection impact by social contexts.

C. Knowledge Management Implications

TABLE II. KNOWLEDGE MANAGEMENT IMPLICATIONS

Knowledge Management Implications	
Consulting Company (knowledge supplier)	Enterprise (knowledge consumer)
<ol style="list-style-type: none"> 1. design practice teams carefully on dependence and perspectives issues 2. join and get perspectives from different communities of practice 3. balance innovation and legitimacy perspective 4. watch social identity issues 5. employ different background employees 	<ol style="list-style-type: none"> 1. understand consulting company or other knowledge sources logics and their perspectives 2. balance absorbing knowledge from different knowledge sources and markets 3. collect more opinions and perspectives from other markets before important R&D or marketing decisions 4. employ different background employees

In this study, the reports represent the large OEM perspectives and powerful groups' interests that influence the government's technology policy. The way of knowledge representation and integration are impacted the institutionalized knowledge markets. Thus, knowledge suppliers and consumers should consider the issues and take strategies to solve problems (see Table 2).

For example, in consulting companies or knowledge suppliers, their managers should consider complexity, relationships, dependence of knowledge boundaries connected with practices while designing and creating reports.

The firms should also consider balance of innovative or legitimacy perspectives under market mechanism in social contexts. The publications or knowledge spanning different communities of practice in knowledge markets also emerge conflicts issues of social identities. The firms should deal with the social identities issues carefully.

The enterprise or knowledge consumers should examine logics of knowledge representation and other perspectives precisely to prevent losing other possible ICT applications or development opportunities.

VII. CONCLUSION

This study illustrates how knowledge as commodity selling in the market using communities of practice perspective. This study elaborates on why and how organizations produce knowledge commodity upon the influence of market mechanism and social contexts. Further, emerging issues such as knowledge integration and dependent relationships, knowledge representation and power in knowledge markets are worthy to address and further

This study is limited to the comparison between practices and teams in one organization. Because of studying one organization, which experienced a particular history and regional location, we are unable to provide a wider understanding of the contexts under which changes of boundary objects might occur.

However, our findings are potentially generalized to other knowledge markets in which knowledge goods creation, exchanged and institutionalized. Future research can conduct inter-organization study or comparisons, and probe into the roles, social identities, meaning, and knowledge representations, competing mechanisms of knowledge goods in various social contexts and knowledge markets.

REFERENCES

- [1] A. Lam, "Tacit Knowledge, Organizational Learning and Societal Institutions: An Integrated Framework," *Organization Studies*, 2000, pp. 487-513.
- [2] J. Lave and E. Wenger, *Situated Learning: Legitimate Peripheral Participation*, New York: Cambridge University Press, 1991.
- [3] J. S. Brown and P. Duguid, "Knowledge and Organization: A Social-Practice Perspective," *Organization Science*, 2001, pp. 198-213.
- [4] M. R. Tagliaventi and E. Mattarelli, "The Role of Networks of Practice, Value Sharing, and Operational Proximity in Knowledge Flows between Professional Groups," *Human Relations*, 2006, pp. 291-319.
- [5] J. Swam, H. Scarbrough, and M. Robertson, "The Construction of 'Communities of Practice' in the Management of Innovation," *Management Learning*, 2002, pp. 477-496.
- [6] E. Einhorn, "Why Taiwan Matters?" *Business Week*, May, 2005.
- [7] H. Karsten, K. Lyytinen, M. Hurskainen, and T. Koskelainen, "Crossing Boundaries and Conscription Participation: Representing and Integrating Knowledge in a Paper Machinery Project", *European Journal of Information Systems*, 2001, pp. 89-98.
- [8] P. R. Carlile and E. S. Rebertisch, "Into the Black Box: The Knowledge Transformation Cycle", *Management Science*, 2003, pp. 1180-1195.
- [9] B. P. Bloomfield and T. Vurdubakis, "Re-presenting Technology: IT Consultancy Reports as Textual Reality Constructions", *Sociology*, 1994, pp. 455-477.
- [10] A. Strauss and J. Corbin, *Basics of Qualitative Research: Grounded Theory Procedures and Techniques*, Sage Newbury Park, CA, 1990.
- [11] P. Franson, "The Market Research Shell Game," *Upside*, 1997, pp. 78-116.
- .

Exploring Statistical Information for Applications-Specific Design and Evaluation of Hybrid XML storage.

Lena Strömbäck, Valentina Ivanova, David Hall

Department of Computer and Information Science

Linköpings Universitet

S-581 83 Linköping, Sweden

lena.stromback@liu.se, valentina.ivanova@liu.se, david.hall@liu.se

Abstract — Modern relational database management systems provide hybrid XML storage, combining relational and native technologies. Hybrid storage offers many design alternatives for XML data and in this paper we explore how to aid the user in effective design of hybrid storage. In particular we investigate how the XML schema and statistical information about the data can support the storage design process. We present an extended version of our tool HShreX that uses statistical information about a data to enable fast evaluation of alternative hybrid design solutions. In addition we show the benefit of the approach by a first evaluation where we discuss how the tool aids in the storage design and evaluation process.

Keywords – *XML, Hybrid XML management, indexing, storage design*

I. INTRODUCTION

The rapid increase in web based applications yields an increasing interest in using XML for representation of data. XML is able to represent all kinds of data ranging from marked-up text, through so called semi-structured data to traditional, well structured datasets. Supporting the flexibility that makes XML appealing is challenging from data management and technical perspectives. Several approaches have been used including native databases and shredding XML documents into relations. In practice, hybrid storage that combines native and relational solutions is of large interest. Hybrid storage is provided by the major relational database vendors (Oracle, IBM DB2 and Microsoft SQL Server). They offer interesting options for storage design where native and relational storage can be used side by side.

Several studies evaluate different solutions for XML management (e.g., [22][24][26][31]). For shredding, it is well known that the choice of translation strategy affects the efficiency (e.g., [5][8][10][12]) while hybrid XML storage, has so far only been studied in a few cases, (e.g., [16][17][27][28]). The above studies discuss a number of features that may have an impact on how to achieve efficient storage; the complexity and regularity of the XML structure; how the data is queried, i.e., the access patterns for different entities in the data set; and the frequency of references to other sources.

In this paper, we further explore these issues by investigating the impact of the application on the performance of the database. The properties we are focusing

on are the XML schema structure and statistical properties of the data set. We first give some further motivation and discuss the goals of our work. This is followed by a discussion of properties and measurements relevant for storage design. We then present a tool that enables fast evaluation and exploration of storage solutions and present a first evaluation to show the feasibility of the tool. The paper is summarized by presenting our future vision. Our long term goal with the work is to present a method that can suggest a set of plausible hybrid storage models for an application.

II. MOTIVATION AND GOALS

Previous work (e.g., [1][3][5][8][23]) have defined efficient shredding methods for XML data into relational databases that result in fast query times. For hybrid storage the situation is more complex where an inappropriate choice of storage design can lead to poor performance [25]. In general automatically shredded relational XML mappings can lead to a rather large and complicated structure of relations. On the other hand, storing entire XML documents natively in XML storage keeps the structure completely intact to the cost of slow access to the data. For hybrid XML storage we have the choice to store parts of the XML structure as relations and other parts as XML and can gain from the benefit of a good data model and relatively fast performance. The design of a good hybrid storage model is complex and dependent on the requirements for the specific application [25].

Exploration and evaluation of alternative solutions is a time consuming task. Methods and tools, to aid the user in design of hybrid storage, and measurements, that could give hints on how to make choices, are of high importance. In a preliminary evaluation we compared the query efficiency with the amount of data stored as XML in the hybrid solution. In our tests, we adopt the shredding principles used in ShreX [1][6] as these principles give a mapping that captures the semantics of a given XML schema for the XML data. To explore hybrid storage we used the extended system HShreX [27][29], which also allows hybrid XML mappings. The general principle behind the mappings of these systems is that complex elements are translated to relational tables. Simple elements and attributes are shredded to a column in their parent table if they occur at maximum once in its parent element. HShreX extends this basic shredding by providing

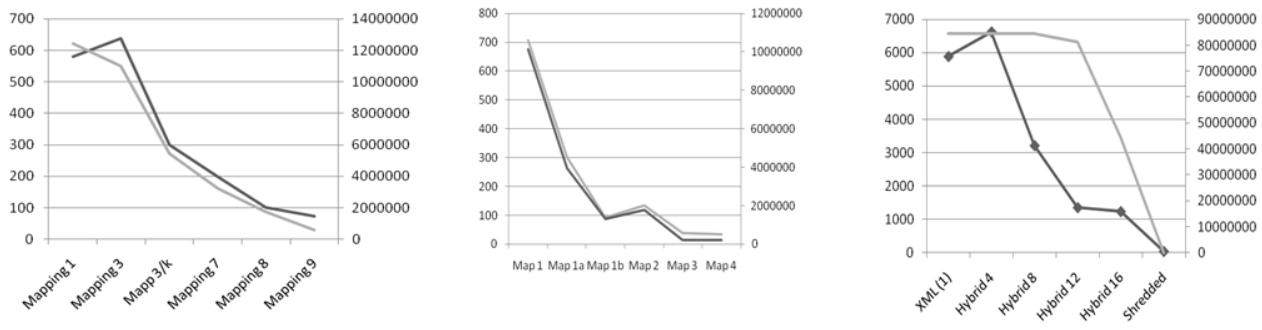


Figure 1. Run times [ms] (black) and data size [bytes] (grey) for PSI-MI (left), UniProt (middle) and Michigan Benchmark (right)

hybrid XML storage, i.e., to allow parts of the structure to be kept as XML in the final database representation. In our study the complexity of the created models varies between one or two relations for the models stored in pure XML to over 100 relations for the fully shredded data models.

The results of these tests are illustrated in Figure 1. The first two graphs show the results for two real data sets from the IntAct [2] and UniProt [30] databases. In this case we can see that the amount of data stored as XML gives a good estimation of the expected query time. For the Michigan Benchmark data [19] the estimation is not as good as for the two other datasets. This means that the amount of data is a good indicator for the performance, but also that further statistics about the data could give us better indicators and aid in effective storage design.

III. AVAILABLE INFORMATION

In principle there are three sources of information that can be used to learn more about the features and storage requirements of a computer application. These are; The general data schema, i.e., the data model; Samples of data to determine how the data model is used and what parts of the data model are in most common use; Samples of queries to determine what kind of queries are often performed for the data. In this work, we will examine how to use the data model and statistical information for a particular dataset.

As shown in the previous section the amount of data

stored as XML is related to the query performance. However, the prediction we get from simply measuring the amount of data is not enough, we also need to collect more detailed information about the structure of the data. In practice, different parts of the XML schema are populated differently in different data sets. The XML schema carries information about the general structure, but, as for relational databases, the schema does not give a full picture of how this structure is instantiated for a particular dataset. We want to capture this information to create an effective hybrid storage model. In previous work [13], where we worked with generated data, we could see that also the amount of data at various positions in the XML file and the structure of this data had an impact on query performance. We wanted to explore this further and collected the following information:

- Overall statistics for the dataset. With this we mean characterizing the general structure of the dataset. For this purpose we use simple measures, such as, the total number of attributes, elements, and levels in the XML. We also collect the number of elements at each level of the dataset to determine the fan out of the data.
- Diversity of the dataset. To get estimations of diversity we collect the number of elements and attributes for each element or attribute string, at which depths they occur and compare those to the number of overall elements. We also collect information on how many unique search paths occur within the data set and the number of their occurrence.
- Detailed information at each position in the file. This is collected by counting the occurrence of element names at each level in the file. For each combination of parent/child node we count how common the child node is for this parent and collect the minimum, maximum and mean number of times this child occurs for the parent.

Our work on generated data has shown that parent/child statistics were of particular interest since this had a large impact on query performance. Figure 2 shows an example of the parent/child statistics. In the XML schema tree we show how common the different child nodes are in the parents.

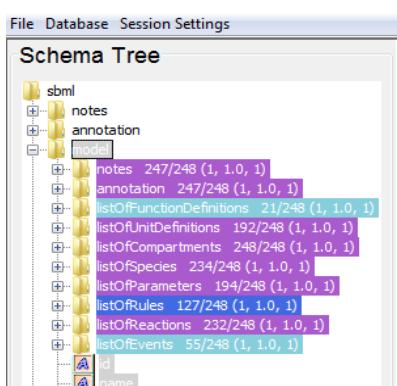


Figure 2. Statistics in HShreX

IV. A TOOL FOR EVALUATION

To allow easy access to the statistics and aid in evaluating storage alternatives we extended our tool HShreX to include this new information. The new version of the tool can be used to create and evaluate different XML storage models. The system analyses an XML schema and represents it as a tree structure, which facilitates its visual perception. The tree structure helps to easily understand and navigate the schema components as well. The relational schema, which the HShreX user can create in a database, is likewise created during the schema analyses. Once the database structures are created, large datasets, which corresponds to the currently parsed schema, can be quickly shredded in the database. Each step starting from the XML schema parsing and ending in datasets loading is logged and available for review in a panel under the main work area.

The relational schema is created following the shredding strategy, mentioned above. The user can alter the data shredding rules using HShreX annotations [27]. In this way, the XML data can be represented in purely native, mixed and shredded storage models. The HShreX annotations provide the opportunity to switch rapidly and flexibly between different storage models, create them in a database and evaluate their performance features.

HShreX's user interface provides three panels, which give more details of the schema elements and their mappings. The first panel lists specific details, such as currently applied HShreX annotations, children elements and attributes and their occurrences, for the currently selected element in the XML schema tree. The second shows HShreX mapping of the selected element or attribute in the tree. The relational tables and their relations are available in figures in the third panel.

In this work, the user interface was extended in two directions – to provide more convenient work with HShreX annotations and to visualize more information for a particular dataset. Figure 3 shows the dialog that facilitates manipulation of HShreX annotations. While navigating in the schema tree, we can open the dialog for the element or the attribute of interest and process its annotations. The dialog provides functionality for adding annotations, updating, i.e., changing values of available annotations and deleting annotations. Since some combinations of annotations for an element or an attribute are not valid, we validate each annotation regarding the already available annotations prior to adding. A useful feature is provided through the “Apply all changes to all elements of this type” button i.e., the currently added/removed annotations will be applied to all elements of this type in the XML schema with a single action. The basic data and the annotations, which apply to the element or the attribute of interest, are listed in the right side of the dialog.

The second improvement in the user interface is orientated towards the statistical information available for a particular dataset. HShreX obtains this information by analyzing a set of sample XML files representing the dataset. Detailed information, for the element or the attribute of interest and its children elements and attributes, is presented

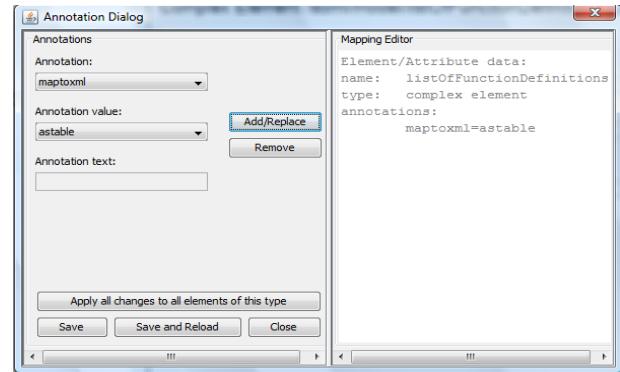


Figure 3. Add/remove annotations dialogue

in the schema tree when a particular dataset is loaded to the database in use. Three different colors are used to facilitate user's perception and to show how many times a particular child node appears under its parent element i.e., different children nodes are colored depending on their frequency of appearance. Thus, the user gets fast and highly useful overview of children nodes and can prioritize his next studies based on this information. The schema tree representation of statistical information aids the user decision on what annotations are appropriate to be used for a particular dataset and helps to construct proper queries with higher efficiency. Further, the statistics can help the user to create indexes and optimize queries.

The other part of the statistical data described in the previous section can be found in “Open Main Statistics” and “Open All Statistics” dialogs under the “File” menu. The statistical data visualized in the schema tree is small, however, our experience have shown that it is the most useful part of the information available for the dataset.

V. A FIRST EVALUATION

In this section, we present the results from the preliminary study of our approach, a more extensive research could be a subject of a future work. To explore the benefit of our tool and the statistical information, we used it to evaluate the performance on the homo sapiens dataset from the REACTOME database [18] and on the BIOMODELS dataset [7], both corresponding to the SBML 2.1 XML schema [14]. The data in the first dataset is spread in depth (the data is stored on many levels) and the data in the second dataset is spread in width (the data is populated almost equally within the dataset).

The statistical information for the dataset of interest becomes available in HShreX when it is loaded in the database in use. The statistics available directly in the HShreX schema tree gives detailed information for the occurrence of the nodes and their parents and present a clear view of data distribution in the particular dataset. This data is presented in the interface as a number pair in the child node name where the first number shows the number of times the child element occurs under its parent and the second shows the number of times the parent element is presented in the dataset at this level. The three numbers, in the parenthesis in

```

Shredded:
SELECT a."id", a."name"
FROM sbml_model_listOfReactions_reaction a,
sbml_model_listOfReactions_reaction_listOfReactants b,
sbml_model_listOfReactions_reaction_listOfReactants_speciesReference c
WHERE a."shrex_id" = b."shrex_pid"
AND b."shrex_id" = c."shrex_pid"
AND c."species" = 'REACT_5251_1_Oxygen';

Native:
SELECT reaction.query( 'for $i in /reaction/listOfReactants/speciesReference
where $i/@species = "REACT_5251_1_Oxygen"
return <Details> {$i/../@id} {$i/./@name} </Details>' ) "data"
FROM sbml_model_listOfReactions_reaction
WHERE reaction.exist('/reaction/listOfReactants/speciesReference
[@species="REACT_5251_1_Oxygen"]') = 1;

```

Figure 4. Sample query for SBML – Query 1

the child node name, show the minimum, the maximum and the average number of times this child occurs for this parent.

Examining the mentioned datasets, using the HShreX interface, we noticed that some of the elements and their parents occur more often than others, thus our research will be more productive if we concentrate on them. Therefore in our examples we have applied the HShreX annotation **maptoxml** to the *reaction* and to the *model* elements in the XML schema. This particular annotation/value combination has been selected in order to force the HShreX application to store these parts of the data as pure XML in the corresponding database. If we do not apply any HShreX annotations the data in the datasets is represented in a shredded storage model (positions 1 and 2 in Figure 6). The HShreX has been forced to represent the data in a hybrid and in a pure native storage models applying the **maptoxml** annotation to the *reaction* (positions 4 to 8) and *model* (positions 10 to 14) elements respectively.

We have chosen two of the major database servers available on the market and set up their options related to the XML data representation in various configurations. Using the database servers XML storage capabilities we are able to store the XML data with or without associating it with corresponding XML schema. The database servers run on HP Proliant DL380 G6 Server with two Intel Xeon E5530 Quad Core HT Enabled processors running at 2.4 GHz (in total 16 logical processors) and 30 GB RAM.

We have created different SQL queries (exemplified in Figure 4 and Figure 5) and executed them against the three

```

Shredded:
SELECT species, shrex_pid, species
FROM sbml_model_listOfReactions_reaction_listOfReactants b,
sbml_model_listOfReactions_reaction_listOfProducts c,
sbml_model_listOfReactions_reaction_listOfReactants_speciesReference d,
sbml_model_listOfReactions_reaction_listOfProducts_speciesReference e
WHERE c."shrex_pid" = b."shrex_pid"
AND b."shrex_id" = d."shrex_id"
AND c."shrex_id" = e."shrex_id"
AND d."species" = 'REACT_5251_1_Oxygen';

Native:
SELECT reaction.query( 'for $react in //reaction,
$rtant in $react/listOfReactants/speciesReference,
$prod in $react/listOfProducts/speciesReference
return <path> {data($rtant/@species)} {data($react/@id)}
{data($prod/@species)} </path>' ) "test"
FROM sbml_model_listOfReactions_reaction
WHERE reaction.exist('//reaction/listOfReactants/speciesReference
[@species="REACT_5251_1_Oxygen"]') = 1;

```

Figure 5. Sample query for SBML – Query 2

storage models and different database configurations. In Query 1, the simpler among both, we retrieve details for a reaction where one of its participants is specified. In the second query, we join details for reactions and reactions to extract participants and products for all reactions. First we executed the two queries using only the homo sapiens dataset. After that we loaded both datasets at the same time and evaluated how the response time changes when the size of the data stored in the database increases. The measured performance can be influenced by other processes running on the server. To reduce this influence, the queries from the figures were executed ten times per condition set, and the averages of the results are presented.

First runs were made without any additional optimization. Based on the statistics, proper XML indices, for each variation of database storage options, were created and the same queries were executed again. Thus, we benefit from the statistical information available for a particular dataset in three ways: we can use the statistics to choose the best place for the HShreX annotations regarding our interests and in this way to switch flexibly and rapidly between different storage models. We are as well able to create proper, for each storage model, indices based on the view of the data distribution in the particular dataset. A final advantage is that we can optimize our SQL queries not only creating indices but rewriting them based on the data

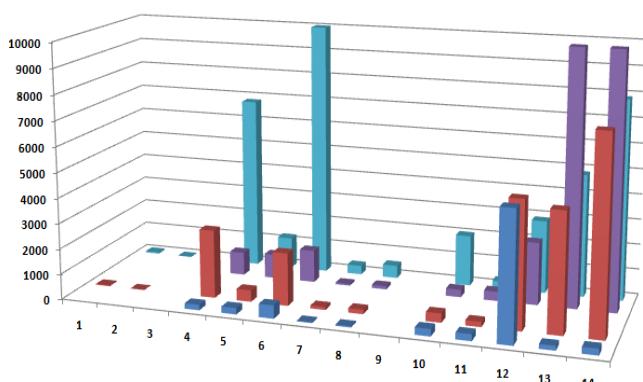
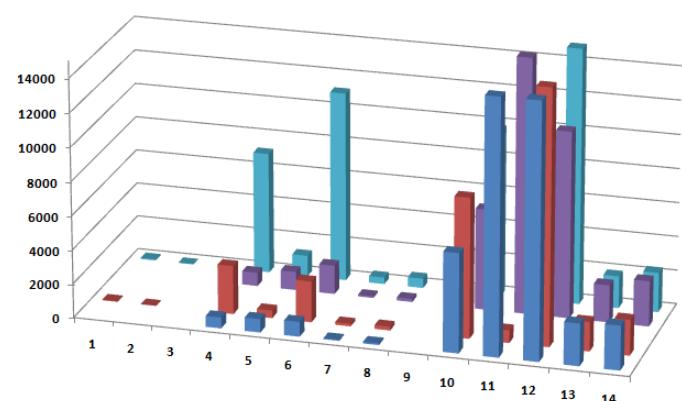


Figure 6. Performance [ms] for Query 1 (left) and Query 2 (right) where: ■ homo sapiens dataset with index, ■ homo sapiens dataset without index, ■ homo sapiens and biomodels datasets with index and ■ homo sapiens and biomodels datasets without index



distribution and complexity.

The results from the two different query executions are shown on Figure 6. The equivalent positions on the 'X' coordinate in both of the charts correspond to equivalent condition sets of database storage options. The results from positions 1 and 2 correspond to a fully shredded storage, positions 4 – 8 correspond to a hybrid storage and positions 10 – 14 correspond to a pure native XML storage. Positions 4 – 8 use the same conditions sets of database storage options as positions 10 – 14, however the HShreX annotation is applied to different elements. As we expected, there is a clear relation between the storage model and the query performance i.e., the execution times are fastest in the shredded storage and slowest in the pure native storage.

Examining the positions 4 – 14 in both result sets we can clearly see that the query performance varies with a different amount for the different database storage options when the size of the data in the database increases. The performance is usually improved when the XML indices are created. It is worth noting that this is not true for position 11 in Query 2 where the performance drops considerably when the index is used. While positions 4 – 8 in the two results sets are comparable, positions 10 – 14 have a lot of differences. Positions 13 and 14 in the first results set have the worst performance among the results for pure native storage while in the second results set they have the best performance. Analyzing positions 13 and 14 in the first result set shows that indices have excellent performance when the size of the data is relatively small and their performance decrease when the data size increases. It is worth noting as well the differences between positions 7, 8 and respectively 13, 14 in the results for Query 1. Positions 7, 13 and 8, 14 respectively have the same database storage options – positions 7 and 8 give the best results while positions 13 and 14 give the worst.

Analyzing the two result sets we can conclude that indices provide better results when used with the hybrid storage than with the pure XML storage. The indices efficiency increases when the size of the data in the hybrid storage increases. During results analysis we need to consider that the results are also affected from the database servers XML storage capabilities and created indices. The benchmark results are influenced from the data distribution in the datasets as well as the SQL queries construction. The statistical data available in HShreX facilitates and aids our decision where to put HShreX annotations and SQL indices and thus HShreX assists us in fast storage construction.

VI. RELATED WORK

The work presented in this article combines ideas from several different areas for XML storage. The first is the work on automatic shredding of XML documents into relational databases by capturing the XML structure or based on the DTD or XML schema for the XML data [1][5][6][8]. The intention with these approaches is to create efficient storage for the XML data. The resulting data model is often hard to understand and is usually hidden from the user via an interface providing automatic query translation of XQuery into the model.

The other related area is hybrid XML storage for relational databases. The vendors offer different underlying representation for the XML type, in some cases it is a byte representation of the XML, in other cases it is some kind of shredding of the XML data [4][10][20][23]. In addition, database vendors provide a number of tools to import XML natively or shred the data into the system. These tools are intended for design of one database solution, thus generation and evaluation of alternative solutions become time consuming.

Interesting work [21] has addressed the question of properties of XML data and generating statistical and comparative measurements of XML datasets. However, this work concentrates on overall measures of properties of the dataset and does not consider the more detailed statistical measurements that we have found most useful in our work.

Other related works are found within database optimization [11][15]. Query optimization can rely on statistics of data and query use for fine tuning their performance [9][12]. However, these statistics are often dependent on the internal database representation instead of based on the original dataset as is necessary for our work. It would be interesting to include these measurements in our work to see whether they could give added value to our indicators.

VII. CONCLUSION AND FUTURE DEVELOPMENT

The first tests of the tool are promising and show that our tool is very useful for aiding in storage design. Using the tools and statistics improves the evaluation process and makes it possible to compare a high number of alternative hybrid database designs. We will continue to use the tool for more extensive evaluations and to refine the method. In particular, we want to compare our set of measurements with the more advanced statistical methods used in [9]. The final goals would be to use the measure to provide suggestions of beneficial hybrid data models for the end user, to further automate the process of storage design. To reach this goal it is crucial to have access to series of data with specific properties to fine tune the indicators and tests. Also for this issue we have made a first solution for generating data with desired properties [13], which can be integrated into our tool.

One bottleneck with our method is that hybrid data models are very complex to query due to the mix of query languages. We are currently using SQL/XML, however, if we consider a user that want to work on the data as if it was XML, this is not feasible. Options are automatic query translations from XQuery to the defined model or to provide a higher level query language for the user.

Another very interesting question is hybrid storage solutions with several DB architectures as a backend, for instance pure native XML databases or specialized databases for graphs or RDF storage. This becomes particularly important for applications where parts of the data contain RDF code or represent graphs as is the case for many system biology standards. We have previously evaluated different combinations [27][28] and would like to include also these options in the HShreX Framework.

ACKNOWLEDGMENT

We acknowledge the financial support from the Center for Industrial Information Technology and the Swedish Research Council. We are also grateful to Juliana Freire for support and fruitful discussions regarding this work and for Mikael Åberg for implementation work on the HShreX tool.

REFERENCES

- [1] S. Amer-Yahia, F. Du, and J. Freire, A Comprehensive Solution to the XML-to-Relational Mapping Problem, Proceedings of the ACM International Workshop on Web Information and Data Management, Nov. 2004, pp. 31-38, doi:10.1145/1031453.1031461.
- [2] B. Aranda et al., The IntAct molecular interaction database in 2010, Nucleic Acids Research, Oct. 2009, pp. 1-7, doi:10.1093/nar/gkp878.
- [3] D. Barbosa, J. Freire, and AO. Mendelzon, Designing Information-Preserving Mapping Schemes for XML, Proceedings of the International Conference on Very Large Databases, Aug.-Sep. 2005, pp. 109-120.
- [4] K. Beyer, F. Özcan, S. Saiprasad, and B. Van der Linden, DB2/XML:Designing for Evolution, Proceedings of the ACM SIGMOD International conference on Management of data, Jun. 2005, pp. 948-952, doi:10.1145/1066157.1066299.
- [5] B. Bohannon, J. Freire, P. Roy, and J. Siméon, From XML Schema to Relations: A Cost-Based Approach to XML Storage, Proceedings of the IEEE International Conference on Data Engineering, Feb.-Mar. 2002, pp. 64-75, doi:10.1109/ICDE.2002.994698.
- [6] F. Du, S. Amer-Yahia, and J. Freire, ShreX: Managing XML Documents in Relational Databases, Proceedings of the International Conference on Very Large Databases, Aug.-Sep. 2004, pp. 1297-1300.
- [7] European Bioinformatics institute <http://www.ebi.ac.uk/biomodels-main/> 25.09.2010.
- [8] D. Florescu and D. Kossmann, Storing and Querying XML Data using an RDMBS, IEEE Data Engineering Bulletin, vol. 22(3), 1999, pp. 27-34.
- [9] J. Freire, JR. Haritsa, M. Ramanath, P.Roy, and J. Siméon, StatiX: making XML count, Proceedings of the ACM SIGMOD International conference on Management of data, Jun. 2002, pp. 181-191, doi:10.1145/564691.564713.
- [10] H. Georgiadis and V. Vassalos, XPath on steroids: Exploiting relational engines for XPath performance, Proceedings of the ACM SIGMOD International conference on Management of data, Jun. 2007, pp. 317-328, doi:10.1145/1247480.1247517.
- [11] G. Gottlob, C. Koch, and R. Pichler, Efficient Algorithms for processing Xpath Queries, ACM Transactions on Database Systems, vol. 30, No 2, Jun. 2005, pp. 444-491, doi:10.1145/1071610.1071614.
- [12] T. Grust, J. Rittinger, and J. Teubner, Why Off-the-Shelf RDMBMSs are Better at Xpath Than You Might Expect, Proceedings of the ACM SIGMOD International conference on Management of data, Jun. 2007, pp. 949-958, doi:10.1145/1247480/1247591.
- [13] D. Hall and L. Strömbäck, Generation of Synthetic XML for Evaluation of Hybrid XML Systems, In: M. Yoshikawa et al. (Eds) Database Systems for Advanced Applications 15th International Conference, International Workshops: GDM, BenchmarX, MCIS, SNSMW, DIEW, UDM, Apr. 2010, Revised Selected Papers. Lecture Notes in Computer Science, vol. 6193, 2010, pp. 191-202, doi:10.1007/978-3-642-14589-6_20.
- [14] M. Hucka et al., The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models, Bioinformatics, vol. 19(4), 2003, pp. 524-531, doi:10.1093/bioinformatics/btg015.
- [15] J. McHugh and J. Widom, Query optimization for XML, Proceedings of the International Conference on Very Large Databases, Sep. 1999, pp. 315-326.
- [16] I. Mlynkova, Standing on the Shoulders of Ants: Towards More Efficient XML-to-Relational Mapping Strategies, Proceedings of the International Workshop on Database and Expert Systems Applications, Sep. 2008, pp. 279-283, doi:10.1109/DEXA.2008.16.
- [17] MM. Moro, L. Lim, and Y-C. Chang, Schema Advisor for Hybrid Relational-XML DBMS, Proceedings of the ACM SIGMOD International conference on Management of data, Jun. 2007, pp. 959-970, doi:10.1145/1247480-1247592.
- [18] Reactome – a curated knowledgebase of biological pathways <http://reactome.org> 25.09.2010.
- [19] L. Runapongsa, JM. Patel, HV. Jagadish, Y. Chen, and S. Al-Khalifa, The Michigan Benchmark: Towards XML Query Performance Diagnostics, Information Systems, vol. 31(2), Apr. 2006, pp. 73-97, doi:10.1016/j.is.2004.09.004.
- [20] M. Rys, XML and relational Management Systems: Inside Microsoft SQL Server 2005, Proceedings of the ACM SIGMOD International conference on Management of data, Jun. 2005, pp. 958-962, doi:10.1145/1066157.1066301.
- [21] I. Sanz, M. Mesiti, G. Gurrini, and RB. Llavori, An entropy based characterization of the heterogeneity of XML collections, Proceedings of the International Workshop on Database and Expert Systems Applications, Sep. 2008, pp. 238-242, doi:10.1109/DEXA.2008.55.
- [22] AR. Schmidt, F. Waas, M. Kersten, MJ. Carey, I. Manolescu, and R. Busse, XMark: A Benchmark for XML Data Management, Proceedings of the International Conference on Very Large Databases, Aug. 2002, pp. 974-985.
- [23] J. Shanmugasundaram, K. Tufte, G. He, C. Zhang, D. DeWitt, and J. Naughton, Relational databases for querying XML documents: Limitations and opportunities, Proceedings of the International Conference on Very Large Databases, Sep. 1999, pp. 302-314.
- [24] L. Strömbäck, Possibilities and Challenges Using XML Technology for Storage and Integration of Molecular Interactions, Proceedings of the International Workshop on Database and Expert Systems Applications, Aug. 2005, pp. 575-579, doi:10.1109/DEXA.2005.154.
- [25] L. Strömbäck and J. Freire, XML Management for Bioinformatics Applications, Computing in Science and Engineering, in press.
- [26] L. Strömbäck and D. Hall, An evaluation of the Use of XML for Representation, Querying, and Analysis of Molecular Interactions, In: T. Grust et. al. (Eds) Current Trends in Database Technology – International Conference on Extending Database Technology 2006 Workshops PhD, DataX, IIDB, IIHA, ICSNW, QLQP, PIM, PaRMA, and Reactivity on the Web, Mar. 2006, Revised Selected Papers. Lecture Notes in Computer Science, vol. 4254, 2006, pp. 220-233, doi:10.1007/11896548_20.
- [27] L. Strömbäck, D. Hall, M. Åberg, and S. Schmidt, Efficient XML data management for systems biology: Problems, tools and future vision, International Journal on Advances in Software, vol. 2(2-3), 2009, pp. 217-233, Invited contribution.
- [28] L. Strömbäck and S. Schmidt, An Extension of XQuery for Graph Analysis of Biological Pathways, Proceedings of the International Conference on Advances in Databases, Knowledge, and Data Applications, Mar. 2009, pp. 22-27, doi:10.1109/DBKDA.2009.16.
- [29] L. Strömbäck, M. Åberg, and D. Hall, HShreX – A Tool for Design and Evaluation of Hybrid XML storage, Proceedings of the International Workshop on Database and Expert Systems Applications, Aug.-Sep. 2009, pp. 417-421, doi:10.1109/DEXA.2009.33.
- [30] The UniProt Consortium The Universal Protein Resource (UniProt), Nucleic Acids Research, vol. 36(1), 2008, pp. D190-D195, doi:10.1093/nar/gkm895.
- [31] BB. Yao, MT. Özsü, and N. Khandelwal, XBench Benchmark and Performance Testing of XML DBMSs, Proceedings of the IEEE International Conference on Data Engineering, Mar. 2004, pp. 621-633.

Transforming XPath Expressions into Relational Algebra Expressions With Kleene Closure

Yangjun Chen

*Dept. Applied Computer Science, University of Winnipeg
515 Portage Ave., Winnipeg, Manitoba, Canada, R3B 2E9
y.chen@uwinnipeg.ca*

Abstract—In the problem of translating XPath expressions into SQL queries, the most challenging part is to find a way to minimize the use of least fixpoint (LFP) operators when a DTD graph contains cycles. In this paper, we address this issue and present a new algorithm to do the task based on the recognition of a kind of DTD graphs, which can be reduced to a single node by contracting nodes into their parents one by one. For this kind of DTD graphs, not only the corresponding relational algebra expressions can be efficiently generated, but the use of LFP operators can also be minimized. For those DTD graphs that are not reducible, we devise a different algorithm which is less efficient than the algorithm for reducible graphs, but more efficient than any existing method.

Keywords: XML, XPath, Query Processing

I. INTRODUCTION

With the widespread of XML both as a document format and as a data exchange format, the interest in querying XML data stored in relational databases has increased. With this comes the need for answering XML queries in a relational database system, by translating XML queries to SQL statements [11, 19, 21]. It is quite different from the prevailing methods for evaluating *twig joins* [7, 8, 9, 10, 11].

Let D be a DTD (Document Type Definition). Let R be a relational schema defined for D by using the shared-inlining technique [24], denoted as a mapping $f: D \rightarrow R$. Denote by \mathcal{D} all the XML documents conforming to D . Denote by \mathcal{R} all the possible relational states of R . Then, the storage of a set of documents conforming to D in $DB(R)$ (a database with the relational schema R) can be considered as a mapping derived from f , denoted as $f_s: 2^{\mathcal{D}} \rightarrow \mathcal{R}$.

Given an XPath expression Q , what we want is to find an equivalent relational algebra expression Q' , which can be evaluated against $DB(R)$, such that for any document $d \in \mathcal{D}$, Q on d can be answered by Q' on $f_s(\{d\})$. That is, the set of nodes selected by Q on T equals the set of tuples selected by Q' on $f_s(\{d\})$. We denote this by

$$Q(T) = Q'(f_s(\{d\})).$$

When a DTD is simply a tree or a DAG (*directed acyclic graph*), a simple translation can be conducted by enumerating all matching paths of the input XPath expression in the DTD, sharing common subpaths, rewriting the paths as relational algebra expressions, and taking a union of all of them [12]. However, when a DTD contains recursive element type definition, the problem becomes

challenging [4, 5]. In this case, the interaction between recursion in the DTD and recursion (*descendant-or-self* axis, represented by ‘//’) in an XPath expression significantly complicates the translation.

In the past decade, a lot of work has been done on querying XML data stored in relational databases such as those discussed in [7, 9, 10, 12, 14, 23]. However, as surveyed in [15], in all these methods, except the strategies proposed in [14, 25], the problem of translating recursive XML queries over recursive DTD is not addressed.

The method discussed in [14] is capable of translating path queries with ‘//’ to a sequence of SQL queries using the SQL’99 recursion operator. However, the SQL queries produced by [14] tend to be large and complicated and cannot be effectively optimized. Also, as pointed out by Fan *et al.* [25], the method is applicable only to a very limited class of path expressions.

In [25], Fan *et al.* proposed a different method. The main idea of this method is to transform an XPath expression to an *extended* XPath expression, in which some variables may be used to represent sub-expressions. In addition, any ‘//’ is replaced with a *Kleene closure*. Given an extended XPath expression, a sequence of relational algebra expressions can be easily created. The time complexity of this process is bounded by $O(|D|^3|Q|\log|D|)$. When translated to an extended XPath expressions, a Kleene closure of the form: E^* corresponds to a sub-expression of the form: A/B in an XPath expression, and E represents all the paths from A to B in the corresponding DTD graph.

However, the generated expressions are also very large with many unnecessary joins involved. For example, for the graph shown in Fig. 1, the regular expression generated by Fan’s algorithm [25] for the path from v_1 to v_1 would be

$$e_0 \cup e_0^* \cup ((e_1 \cup e_0^* \cdot e_1) / (e_4 \cdot e_0^* \cdot e_1)^*) \cdot (e_4 \cup e_4 \cdot e_0^*).$$

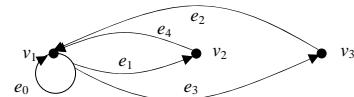


Fig. 1. A directed graph

But the minimized regular expression for this is $(e_0 \cup e_2)^*$. In the Appendix, we will make a sample trace to show how Fan’s algorithm [25] works in generating such a complicated expression over the above graph.

It is obvious that the Kleene closure is a very costly operation. It dominates the whole query evaluation time. So,

it is necessary to reduce the size of E such that as few joins as possible are involved.

In this paper, we propose a new algorithm to mitigate the problem to some extent. We will recognize a class of DTD graphs G , for which a reduction sequence of nodes: $v_1, v_2, \dots, v_n = r$ can be found such that G can be reduced to a single node r , where n is the number of G 's nodes. For this kind of DTD graphs, we can create a relational algebra expression in $O(m\log n)$ time, where m is the edges of G . More importantly, we can always find a way to generate minimized relational algebra expressions for them. For those non-reducible graphs, we propose a different algorithm. Although it is less efficient than the algorithm for the reducible graphs, it is more efficient than any existing strategies.

The paper contains five sections. In Section 2, we review DTDs, XPath expressions, and schema-based mapping from XML to relations. In Section 3, we concentrate on the recursion in XPath expressions. Section 4 is devoted to the general process for transforming XPath queries to relational algebra expressions. Finally, the paper concludes in Section 5.

II. BASIC CONCEPTS

In this section, we review DTDs and XPath expressions, as well as the XML data storage in relational databases to provide a discussion background.

- DTD

Abstractly, an XML DTD can be considered as a triple $\langle H, S, r \rangle$, where H is a set of element types (corresponding to element tag names); r is the root type; and S is a set of rules defining the types in H . That is, for any type A in H , $S(A)$ (the definition of A) is an expression:

$$\beta ::= \varepsilon | B | \beta, \beta | (\beta | \beta) | \beta^*,$$

where ε is the empty word, B represents a type in H (referred to as a subelement or child type of A), and '|', ',' and '*' denote disjunction, concatenation, and the Kleene star, respectively. We refer to $A \rightarrow S(A)$ as the production of A .

We will represent a DTD D as a graph, called the DTD graph of D and denoted by G_D , as done in [24]. In G_D , each node stands for a distinct element type and each edge for a parent/child relationship. In addition, an edge (A, B) is marked with '*' if B is enclosed in a definition of A with the form: β^* .

As an example, see the DTD graph shown in Fig. 2, representing a DTD: $\langle H, S, \text{dept} \rangle$ with

$$H = \{\text{dept}, \text{course}, \text{cno}, \text{title}, \text{time}, \text{prereq}, \text{takenBy}, \text{taughtBy}, \text{professor}, \text{pno}, \text{pname}, \text{teaching}, \text{student}, \text{sno}, \text{sname}, \text{qualified}\}, \text{and}$$

S defined as follows:

$$\begin{aligned} \text{dept} &\rightarrow \text{course}^* \\ \text{course} &\rightarrow \text{cno}, \text{title}, \text{time}, \text{prereq}, \text{takenBy}, \text{taughtBy} \\ \text{prereq} &\rightarrow \text{course}^* \\ \text{takenBy} &\rightarrow \text{course}^* \\ \text{taughtBy} &\rightarrow \text{professor}^* \\ \text{student} &\rightarrow \text{sno}, \text{sname}, \text{qualified} \end{aligned}$$

$$\begin{aligned} \text{qualified} &\rightarrow \text{course}^* \\ \text{professor} &\rightarrow \text{pno}, \text{pname}, \text{teaching} \\ \text{teaching} &\rightarrow \text{course}^* \end{aligned}$$

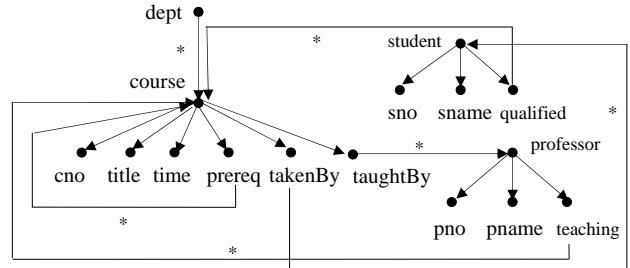


Fig. 2. A DTD graph

In the above DTD graph, we handle each attribute as a primitive element type for simplicity. But it obviously does not lose any generality.

A DTD is recursive if it has an element type that is defined (directly or indirectly) in terms of itself. When represented as a graph, it will contain a few nested and overlapping cycles. So the DTD shown in Fig. 2 is recursive.

- XPath expressions

XPath [6] is a popular language for querying XML data. It has been used in many XML applications and in some other languages for querying and transforming XML data, such as *XQuery* and *XSLT*. In this paper, we address a practical fragment of XPath, in which each *path* in a *predicate* can be compared with a constant, but not with another *path*, given as below:

$$\begin{aligned} p &::= . | A | * | p/p | p//p | p[q] | \\ q &::= p | p \delta c | \neg q | q \vee q | q \wedge q \\ \delta &::= '=' | '!=' | '>' | '>=' | '<' | '<=' \end{aligned}$$

where '.', A , and '*' denote the *self-axis*, a type (element tag name) and a wild card, respectively. '/' and '//' are *child-axis* and *descendant-or-self-axis*, respectively; and $[q]$ is a predicate (also referred to as a *qualifier*), in which c is a constant and δ represents a comparison relation. For example, the following XPath expression

/dept/course[title = 'XML' or
(\neg(time = 2008) and prereq = 'CS2201')]//professor
selects the professor who taught a course either with title
'XML' or with the prerequisite 'CS2201' but not in 2008.
Such an XPath expression can be represented as a tree with
five kinds of nodes: axis-tag nodes (*at-node*), logical-AND
nodes (\wedge -node), logical-OR nodes (\vee -node), logic-negation
node, and constant node:

- *at-node*: An axis-tag node in the tree stands for one location step. It has the content /*tag* or //*tag*.
- \wedge -node: A logical-AND node connects two or more child subtrees with AND logic.
- \vee -node: A logical-OR node connects two or more child subtrees with OR logic.
- \neg -node: A logical-negation node negates the result of its unique subtree.

- **C-node:** A constant node is a value of the form: $= c$, $\neq c$, $< c$, $> c$, $\leq c$, or $\geq c$.

For example, the above XPath expression can be represented as a tree shown in Fig. 3.

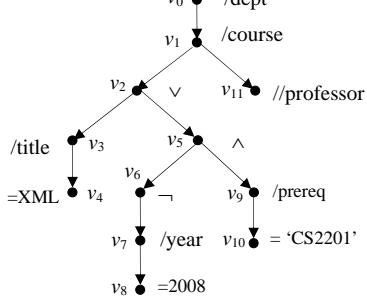


Fig. 3. A tree representing an XPath expression

For any *at-node* v , we use $v.axis$ and $v.tag$ to refer to the axis ('/' or '//') and the tag appearing in v , respectively. In addition, we define some operations on query tree nodes:

$children(v)$ – returns all child nodes of v ;

$parent(v)$ – returns the parent of v ;

$atChildren(v)$ – returns a set of at-nodes in the subtree rooted at v , which are reachable without traversing through other *at-nodes*.

$atParent(v)$ – returns the nearest ancestor *at-node* of v .

For example, given the query tree in Fig. 3, we have $children(v_1) = \{v_2, v_{11}\}$, $parent(v_5) = v_2$, $atChildren(v_1) = \{v_3, v_7, v_9, v_{11}\}$, $atParent(v_9) = v_1$.

- Mapping a DTD into a relation schema

In order to store a set of XML documents (conforming to a certain DTD D) in a relational database, we will first establish a map $f: D \rightarrow R$ from D to a relational schema R . To this end, we will first remove all the edges marked with '*' in G_D , dividing it into several node-disjoint components: G_1, \dots, G_k . Each G_j is then mapped to a relation schema R_j in R , which has three attributes: *ID* (identifier of elements), *P* (parent of the current element) and *V* (for the values of all the other attributes). If G_j has more than one incoming edges, we will use a *parentCode* attribute [24] to distinguish among different parents.

From f , one can easily derive a data mapping $f_s: 2^D \rightarrow \mathcal{R}$, representing the storage of a set of documents in $DB(R)$. Let d be a document conforming to D . Then, in $f_s(\{d\})$ (the database storing d), a tuple (id, p, v) in a relation with a certain schema R_A represents an element in d with its identifier equal to id , its parent element identifier to p , and all its attribute values represented by v . For example, the DTD shown in Fig. 1 can be mapped to four relation schemas: R_d , R_c , R_p , and R_s , representing *dept*, *course*, *professor*, and *student*, respectively. These four relation schemas are connected as shown in Fig. 3(a). In Fig. 3(b), we show a sample database, in which for each relation only values for *ID* and *P* are displayed. (See [24] for a detailed discussion.)

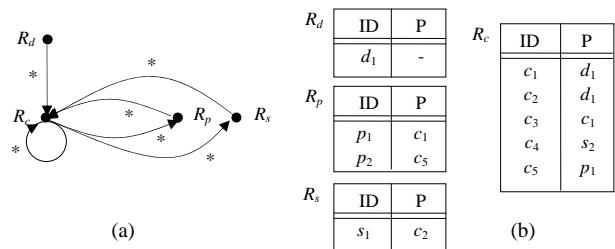


Fig. 4. A tree representing an XPath expression

III. ON THE RECURSION IN XPATH

The most difficult issue in translation of XPath expressions is the treatment of // -axis. In this section, we mainly address this problem. The discussion of a general process for the transformation is shifted to the next section.

A. Reducible subgraphs

Consider an XPath query $A//B$ over a DTD D . This query, when evaluated at an A -element in a document T conforming to D , is to find all B -elements which are the descendants of the A -element. To do this, Fan *et al.* [25] proposed an algorithm to create a sequence of extended XPath expressions to represent all the paths connecting A to B . (An extended XPath is a regular XPath expression [18] with variables being used.) As shown in the introduction, an extended XPath expression generated by Fan *et al.* [25] can be very large.

To mitigate this problem, we recognize a class of graphs, for which not only the corresponding relational algebra expressions can be efficiently created, but the use of LFP operations is minimized.

First, we notice that what we want is to produce an expression for a graph containing all the paths from a node α to another node β in a certain graph G . We call such a graph an $\alpha\beta$ -graph, denoted as $G[\alpha, \beta]$. Obviously, every node in $G[\alpha, \beta]$ is reachable from α .

Definition 1 An $\alpha\beta$ -graph $G[\alpha, \beta]$ is reducible if it can be reduced to a graph consisting of a single node by means of the following transformations:

O_1 (Remove a loop): If e is an edge such that $head(e) = tail(e)$, delete e . (Note that for an edge $e = (a, b)$, $head(e) = a$ and $tail(e) = b$.)

O_2 (Remove a node): If $u \neq \alpha$ is a node such that all edges e with $tail(e) = u$ have $head(e)$ being a same node v , contract u into v by deleting u and all edges from v to u , and converting any edge e with $head(e) = w$ into an edge e' with $head(e') = v$ and $tail(e') = tail(e)$. (We remark that v may be connected to u by multiple edges.) \square

As an example, consider the subgraph $G[R_c, R_p]$ of the graph shown in Fig. 4(a). It can be reduced as shown in Fig. 5.

From Fig. 5, we can see that in each step we remove some loops and then contract a node u ($\neq \alpha$) into another node v if v is the unique parent of u . This process continues until only one node is left.

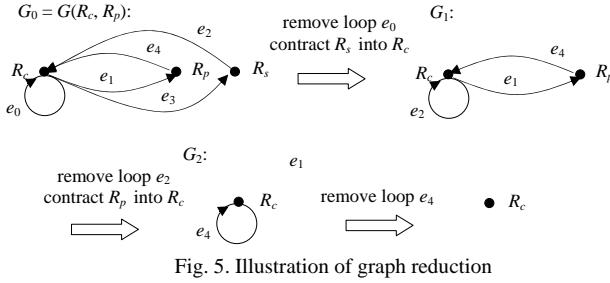


Fig. 5. Illustration of graph reduction

If in a certain step we cannot find a node ($\neq \alpha$) which has only one parent, the process gets stuck. Then the corresponding $\alpha\beta$ -graph is *non-reducible*. Since for the $\alpha\beta$ -graph in the graph shown in Fig. 4(a) the reduction can always be conducted, it is reducible.

To check whether an $\alpha\beta$ -graph is reducible, we do the following operation repeatedly.

1. Remove all the loops.
2. Check each u in the $\alpha\beta$ -graph to see whether O_2 can be applied to it. If it is the case, contract it.

Obviously, this process requires $O(n^2)$ time, where n is the numbers of the nodes of the $\alpha\beta$ -graph.

From the above discussion, we can see that for a reducible $\alpha\beta$ -graph a reduction sequence of nodes: $v_1, v_2, \dots, v_n = \alpha$ can be found such that the $\alpha\beta$ -graph can be reduced to α by removing v_i in the sequence. For convenience, we call α the root of the $\alpha\beta$ -graph. For example, for the $G[R_c, R_p]$ shown in Fig. 5, the reduction sequence of the nodes is: R_s, R_p, R_c . Its root is R_c .

Accordingly, we also get a sequence of graphs: G_0, G_1, \dots, G_{n-1} such that G_0 is the original $\alpha\beta$ -graph and $G_i = G_{i-1}/\{v_i\}$ for $i > 0$ (see Fig. 5 for illustration). For an edge $e \in G_i$, we use $head_i(e)$ and $tail_i(e)$ to represent its head and tail in G_i , respectively. We notice that for the same edge e appearing in G_i and G_j with $i \neq j$, it is possible that $head_i(e) \neq head_j(e)$. For instance, in G_0 (see Fig. 5), $head_0(e_2) = R_s$. But in G_1 , $head_1(e_2) = R_c$. However, for any e , if it appears in G_0, G_1, \dots, G_i for some i , we must have $tail_0(e) = tail_1(e) = \dots = tail_i(e)$.

In the graph reduction process, we can associate each node v with three data structures to facilitate the creation of relational algebra expressions:

loop(v): a set of edges such that for each e in the set there exists G_i for some i such that we have $head_i(v) = tail_i(v)$.

non-loop(v): a set of edges, along which v is contracted into another node. (Remember that we may have multiple edges in a graph.)

contractor(v): a node, into which v is contracted.

Since each node has only one contractor, the contraction process can be represented by a tree (called a *contraction spanning tree* and denoted as *CST*), in which there is an edge from v to u if u is contracted into v .

Example 1 In the graph reduction process shown in Fig. 5, a set of data structures will be constructed, as shown in Fig. 6(a). Fig. 6(b) shows the corresponding CST tree.

	<i>loop(v)</i>	<i>non-loop(v)</i>	<i>contractor(v)</i>
R_s	e_3	R_c	(a)
R_p	e_1	R_c	
R_c	e_0, e_2, e		

Fig. 6. Data structures and contraction tree

In order to generate an expression for a reducible $\alpha\beta$ -graph, we explore the corresponding CST tree bottom-up. During the traversal of the CST, for each encountered node v , the associated data structures are used to generate an expression for it, which is then utilized to create an expression representing all the paths from the root to v .

Algorithm *gen-expression*(T) (* T is a CST.*)

begin

1. search T bottom-up;
 2. **for** each encountered node v **do**
 3. { $E_v \leftarrow \phi, Q_v \leftarrow \phi$
 4. **for** each $e \in non-loop(v)$ **do**
 5. { $E_v \leftarrow E_v \cup R_{head_0(e)} \bowtie R_{tail_0(e)}$; }
 6. **for** each $e \in loop(v)$ **do**
 7. { let $v_1 \rightarrow \dots \rightarrow v_k$ be the path from v to $head_0(e)$ in T ;
 8. $Q \leftarrow E_{v_1} \bowtie \dots \bowtie E_{v_k}$;
 9. $Q_v \leftarrow Q_v \cup Q \bowtie R_{head_0(e)} \bowtie R_{tail_0(e)}$;
 10. }
 11. $E_v \leftarrow E_v \bowtie Q_v^*$;
 12. }
 13. **for** each node v **do**
 14. { let $u_1 \rightarrow \dots \rightarrow u_j$ be the path from α to v in T ;
 15. $E_{\alpha-v} \leftarrow E_{u_1} \bowtie \dots \bowtie E_{u_j}$;
 16. }
- End**

The algorithm works in two phases. The first phase consists of lines 1 - 12. The second phase consists of lines 13 - 16. In the first phase, we create an expression for each node v . In the second phase, for each node v , the expression representing all paths from the root to it is generated by using the expressions generated in the first phase.

Example 2 Applying the above algorithm to the CST tree and the data structures shown in Fig. 6, we will generate the following expressions step by step.

first phase:

Step 1: access $R_s. loop(R_s) = \phi$. $non-loop(R_s) = \{e_3\}$.

$$E_s = R_c \bowtie R_s.$$

Step 2: access $R_p. loop(R_p) = \phi$. $non-loop(R_p) = \{e_1\}$.

$$E_p = R_c \bowtie R_p.$$

Step 3: access $R_c. loop(R_p) = \{e_0, e_2, e_4\}$.

$$head_0(e_0) = R_c. head_0(e_2) = R_s. head_0(e_4) = R_p.$$

$$head_0(e_3) = R_{pub}.$$

$$E_c = (R_p \bowtie R_p \cup$$

$$(R_c \bowtie R_s) \bowtie (R_s \bowtie R_c) \cup$$

$$(R_c \bowtie R_p) \bowtie (R_p \bowtie R_c))^* \\ = (R_p \cup R_c \bowtie R_s \cup R_c \bowtie R_p)^*$$

second phase:

Step 4: $E_{c-s} = E_c \bowtie E_s$

Step 5: $E_{c-p} = E_c \bowtie E_p$

Step 6: $E_{c-c} = E_c \bowtie E_c = E_c$ \square

Proposition 1 Let v be a node in a reducible $\alpha\beta$ -graph. Then the expression $E_{\alpha-v}$ produced by *gen-expression()* exactly represents all paths from α to v .

Proof. We use $\chi(E_{\alpha-v})$ to represent a set containing all the paths represented by $E_{\alpha-v}$. We prove the proposition by induction on the length of the path p from α to v in the corresponding CST tree T .

Basis. When $|p| = 1$, α is the contractor of v . $E_{\alpha-v} = E_\alpha \bowtie E_v$. The proposition holds.

Induction step. Assume that when $|p| = k$ the proposition holds. We consider the case that $|p| = k + 1$. Let v' be the contractor of v . Then the length of the path from α to v' is k . According to the induction assumption, $E_{\alpha-v'}$ exactly represents all paths from α to v' . Since the $\alpha\beta$ -graph is reducible, all paths reaching v must go through v' . So the expression should be $E_{v'} \bowtie E_v$. It is exactly done by the algorithm. \square

B. Non-reducible subgraphs

If an $\alpha\beta$ -graph $G[\alpha,\beta]$ is not reducible, then in the reduction process we will reach a graph G' , in which we are not able to find a node ($\neq \alpha$) that has only one parent.

Let v_1, v_2, \dots, v_j ($j < n$) be the node sequence removed from $G[\alpha,\beta]$ in the incomplete reduction process. Let G_0, G_1, \dots, G_{j+1} be the corresponding graph sequence such that $G_0 = G[\alpha,\beta]$, $G_{i+1} = G_i/\{v_i\}$ ($i = 0, \dots, j$), and G_{j+1} cannot be reduced any more. We call G_{j+1} a remaining graph. Let r_1, \dots, r_l be those nodes in G_{j+1} such that into each of them some v_i ($1 \leq i \leq j$) is contracted. Then, we will construct l CST trees: T_1, \dots, T_l in the same way as discussed in the previous subsection. Each T_i is rooted at r_i ($1 \leq i \leq l$). Assume that β is a node in some T_k ($1 \leq k \leq l$). Then, we can construct an expression $E_{r_k-\beta}$ representing all paths from r_k to β , as discussed in 3.1.

As an example, consider an $G[\alpha,\beta]$ shown in Fig. 7(a) with $\alpha = v_2$ and $\beta = v_7$.

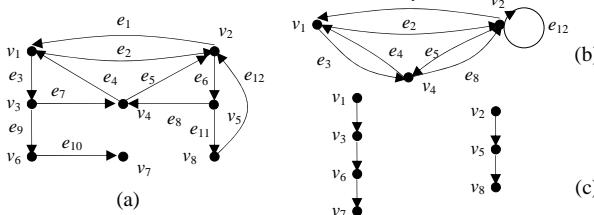


Fig. 7. $\alpha\beta$ -graph, remaining graph, and CST trees

For this graph, we can find a contraction sequence of nodes: v_7, v_6, v_3, v_8, v_5 , and a series of contracted graphs: G_0

$= G[\alpha,\beta]$, $G_1 = G_0/\{v_7\}$, $G_2 = G_1/\{v_6\}$, $G_3 = G_2/\{v_3\}$, $G_4 = G_3/\{v_8\}$, $G_5 = G_4/\{v_5\}$. We show G_5 in Fig. 7(b). It is a remaining graph and cannot be reduced anymore. In G_5 , special attention should be paid to v_1 and v_2 . They are the contractors of v_3 and v_5 , respectively. So, two CST trees will be constructed for the removed nodes as shown in Fig. 7(c). The expression representing all the paths from v_1 to v_7 is easy to compute: $E_{v_1-v_7} = v_7 \bowtie v_6 \bowtie v_3 \bowtie v_1$.

In a next step, we need to calculate $E_{\alpha-r_k}$ over the remaining graph, and produce an expression $E_{\alpha-r_k} \bowtie E_{r_k-\beta}$ as the final result.

However, $E_{\alpha-r_k}$ cannot be calculated in the same way as an expression over a reducible graph. A different algorithm has to be devised. In the following, we discuss this algorithm in detail. Its time complexity is the same as Fan's algorithm [25]. But much less computation will be conducted.

Let G' be a remaining graph. We number its nodes from 1 to n' , where n' is the number of the nodes in G' . Our purpose is to produce a set of expressions of the form E_{i-j} with each representing all paths from i to j , from which $E_{\alpha-r}$ can be created, where r is the root of some CST tree, in which β appears.

So the algorithm works in two phases. In the first phase, we create necessary E_{i-j} 's. In the second phase, we generate $E_{\alpha-r}$.

Procedure phase-1(G')

begin

```

1.   for  $i = 1$  to  $n$  do
2.     {   for  $j = 1$  to  $n$  do
3.       {   if  $e = (i, j)$  is an edge in  $E$  then  $E_{i-j} \leftarrow i \bowtie j$ ;
4.         else  $E_{i-j} \leftarrow \emptyset$ ;
5.       }
6.     }
7.   for  $k = 1$  to  $n$  do
8.     {    $E_{k-k} \leftarrow E_{k-k}^*$ ;
9.       for  $i = k + 1$  to  $n$  do
10.      {   if  $E_{i-k} \neq \emptyset$  then  $E_{i-k} \leftarrow E_{i-k} \bowtie E_{k-k}$ ;
11.        for  $j = k + 1$  to  $n$  do
12.          {   if  $E_{i-j} \neq \emptyset$  then  $E_{i-j} \leftarrow E_{i-j} \cup E_{i-k} \bowtie E_{k-j}$ ;
13.          }
14.        }
15.      }
end
```

Example 3 In this example, we trace the computation when the above algorithm is applied to the graph shown in Fig. 7(b). In the process, the three nodes v_1 , v_2 , and v_4 in the graph are numbered with 1, 2, and 3, respectively. Besides, we use \cdot and e_i to represent respectively \bowtie and $head(e_i) \bowtie tail(e_i)$ for simplicity. We also use I to represent an *identity relation* such that for any relation R we have $I \bowtie R = R \bowtie I = R$. Finally, for a Kleene operation R^* , if R is \emptyset or I , R^* is defined to be I .

Initialization (lines 1 – 5):

$$E^{(0)} = \begin{bmatrix} \phi & e_1 & e_4 \\ e_2 & e_{12} & e_5 \\ e_3 & e_8 & \phi \end{bmatrix}$$

First iteration ($k = 1$):

$$E^{(1)} = \begin{bmatrix} I & e_1 & e_4 \\ e_2 & e_{12} \cup e_2 \cdot e_1 & e_5 \cup e_2 \cdot e_4 \\ e_3 & e_8 \cup e_3 \cdot e_1 & e_3 \cdot e_4 \end{bmatrix}$$

Second iteration ($k = 2$):

$$E^{(2)} = \begin{bmatrix} I & e_1 & e_4 \\ e_2 & (e_{12} \cup e_2 \cdot e_1)^* & e_5 \cup e_2 \cdot e_4 \\ e_3 & (e_8 \cup e_3 \cdot e_1) \cdot (e_{12} \cup e_2 \cdot e_1)^* & e_3 \cdot e_4 \cup (e_8 \cup e_3 \cdot e_1) \cdot (e_{12} \cup e_2 \cdot e_1)^* \cdot e_5 \cup e_2 \cdot e_4 \end{bmatrix}$$

Third iteration ($k = 3$):

$$E^{(3)} = \begin{bmatrix} I & e_1 & e_4 \\ e_2 & (e_{12} \cup e_2 \cdot e_1)^* & e_5 \cup e_2 \cdot e_4 \\ e_3 & (e_8 \cup e_3 \cdot e_1) \cdot (e_{12} \cup e_2 \cdot e_1)^* \cdot (e_3 \cdot e_4 \cup (e_8 \cup e_3 \cdot e_1) \cdot (e_{12} \cup e_2 \cdot e_1)^* \cdot e_5 \cup e_2 \cdot e_4)^* & \end{bmatrix}$$

□

Concerning the above algorithm, we have the following proposition.

Proposition 2 After the execution of *phase-1*, the following statements are true:

- i) $E_{i,j}$ for $i \geq j$ is an expression representing exactly the paths from i to j which contain no intermediate node larger than j .
- ii) $E_{i,j}$ for $i < j$ is an expression representing exactly the paths from i to j all of whose intermediate nodes are smaller than i .

Proof. Straightforward by induction on k . □

In terms of this proposition, we design the second phase to generate $E_{\alpha,r}$.

If $\alpha \geq r$, for all those joins of the form $E_{\alpha,j} \bowtie E_{j,r}$, which should be included in $E_{\alpha,r}$, j should be larger than r . If $\alpha < r$, j should be larger than or equal to α .

According to the above analysis, we give the following procedure.

Procedure *phase-2(E)* (* E contains all the expressions produced in *phase-1*.*)

Begin

1. **If** $\alpha \geq r$ **then** $j \leftarrow r + 1$;
 2. **else** $j \leftarrow \alpha$;
 3. $E_{\alpha,r} \leftarrow E_{\alpha-r} \cup (E_{\alpha,j} \bowtie E_{j,r}) \cup \dots \cup (E_{\alpha-n} \bowtie E_{n-r})$;
- end**

For example, the expression representing all paths from v_2 to v_1 in the graph shown in Fig. 7(b) is

$$\begin{aligned} & E_{2-1} \cup (E_{2-2} \bowtie E_{2-1}) \cup (E_{2-3} \bowtie E_{3-1}) \\ & = e_2 \cup (e_{12} \cup e_2 \cdot e_1)^* \bowtie e_2 \cup (e_5 \cup e_2 \cdot e_4) \bowtie e_3. \end{aligned}$$

Proposition 3 After the execution of *phase-2*, $E_{\alpha,r}$ is an expression representing exactly the paths from α to r .

Proof. In terms of Proposition 2, $E_{\alpha,r}$ generated by *phase-1* represents only those paths from α to r , which contain no node larger than α or larger than r , depending on whether $\alpha \geq r$ or $\alpha < r$. By *phase-2*, the missing sub-expressions are calculated and included in the final expression. □

IV. GENERAL PROCESS

In this section, we describe a general process to transform an XPath expression to a relational algebra expression.

As shown in Section 2, an XPath expression Q , can always be represented as a tree T_Q . Then, given a query tree, what we need to do is to construct an expression from T_Q . Our method works in three steps.

In the first step, we transform each node v of the $//B$ node to a node of the form $/E$ such that E may contain Kleene closures. It is done as follows.

- If B is an attribute name, we will first find the relation name C , in which B appears. Then, find $atParent(v).tag$ and construct an $\alpha\beta$ -graph $G[atParent(v).tag, C]$. Assume that the expression created for the graph is E . We will replace v with an edge (u, u') such that $u = '/E'$ and $u' = '/B'$ and the children of v become the children of u' .
- If B is a relation name, we will construct an expression E for $G[atParent(v).tag, B]$ and replace v with ' $/E$ '.

In this way, we transform T_Q to T_Q' which does not contain any ' $//$ '.

In the second step, we mainly handle attributes. For any *at-node* v of the form $/tag$ with tag being an attribute in some relation R , we will find its child u if it exists. We distinguish two cases of u .

- u is a logic node ' \wedge ' or ' \vee '. In this case, we will find all children of u . Each of them must be a constant node of the form δc , where c is a constant and δ is $=$, $!=$, $>$, \geq , $<$ or \leq . Let $\delta_1 c_1, \dots, \delta_k c_k$ be the children of u . If $u = \wedge$, construct a selection operation

$$\sigma_{tag \delta_1 c_1 \wedge \dots \wedge tag \delta_k c_k}(R).$$

Otherwise, construct

$$\sigma_{tag \delta_1 c_1 \vee \dots \vee tag \delta_k c_k}(R).$$

Substitute it for v .

- u is a constant node δc . Replace v with $\sigma_{tag \delta c}(R)$.

In this way, we transform T_Q' to T_Q'' which does not contain any attribute name.

Now we have only logic nodes and *at-nodes* with relation names in T_Q'' . (See Fig. 8 for illustration.)

In the third step, we will search T_Q'' bottom-up. Let v be the node currently encountered. The following operations will be performed.

1. If v is a leaf node, return v .
2. If v is a non-leaf node, it will be checked as follows.
 - $v = \vee$: Let v_1, \dots, v_k be the children of v . Let F_i be the relational algebra expression F_i for $Q[v_i]$ (subtree rooted at v_i) ($i = 1, \dots, k$). Return $F_1 \cup \dots \cup F_k$.
 - $v = \wedge$: Let v_1, \dots, v_k be the children of v . Let F_i be the relational algebra expression F_i for $Q[v_i]$ ($i = 1, \dots, k$). Return $F_1 \cap \dots \cap F_k$.
 - $v = \neg$: Return $R - F$, where R is the relation name in its *atParent*, and F is the expression created for its unique child.

$v = '/tag'$: Let v_1, \dots, v_k be the children of v . Let F_i be the relational algebra expression F_i for $Q[v_i]$ ($i = 1, \dots, k$). If ' tag ' appears in at least one F_i , return $F_1 \bowtie \dots \bowtie F_k$. Otherwise, return $tag \bowtie F_1 \bowtie \dots \bowtie F_k$.

Example 4 Applying the above process to the query tree shown in Fig. 3, we will get a tree shown in Fig. 8 after the first two steps.

After the third step, we will get the following expression:

$$\text{dept} \bowtie (\sigma_{\text{title}=\text{XML}}(\text{course}) \cup ((\text{course} - \sigma_{\text{year}=2008}(\text{course})) \cap \sigma_{\text{prereq}=\text{CS2001}}(\text{course})) \bowtie E. \quad \square$$

The above expression can be evaluated in any relational database system which supports the LFP operation. But such an expression should be optimized. This can be done by using the standard techniques [26].

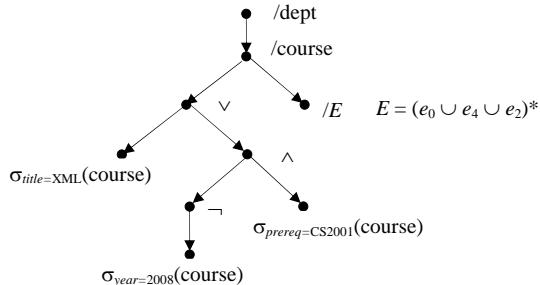


Fig. 8. Illustration for tree transformation

V. CONCLUSION

In this paper, a new method is proposed for transforming unique parents. For this kind of graphs, not only the expressions can be efficiently generated, but the use of the LFP operators can be minimized. For a non-reducible graph XPath expressions to relational algebra expressions. The main idea behind it is to recognize a class of DTD graphs, which can be reduced by contracting nodes into their respective, we divide it into two parts: a reducible part and a non-reducible part, and create expressions for them separately. In this way, the use of the LFP operators can also be dramatically decreased. In addition, a theoretical comparison of our method with Fan's algorithm is conducted, showing that Fan's algorithm is in essence a brute-force algorithm, by which no attention is paid to the structure of DTD graphs. So it cannot be efficient, especially for the reducible graphs.

REFERENCES

- [1] R. Agrawal and P. Devanbu. Moving selections into linear least fixpoint queries. In *ICDE*, 1988.
- [2] F. Bancilhon, D. Maier, Y. Sagiv, and J. Ullman. Magic sets and other strange ways to implement logic programs. In *PODS*, 1986.
- [3] C. Beeri and R. Ramakrishnan. On the power of magic. *J. Log. Program.*, 10, 1991.
- [4] BIOML. BIOPolymer Markup Language http://xml.coverpages.org/_BIOML-XML-DTD.txt.
- [5] Y. Chen. Magic Sets and Stratified Databases, *Int. Journal of Intelligent Systems*, John Wiley & Sons, Ltd., Vol. 12, No. 3, March 1997, pp. 203-231.
- [6] Y. Chen, On the Bottom-up Evaluation of Recursive Queries. *Int. Journal of Intelligent Systems*, John Wiley & Sons, Ltd., Vol. 11, No. 10, Oct. 1996, pp. 807-832.
- [7] Y. Chen, An Efficient Streaming Algorithm for Evaluating XPath Queries, in: *Proc. of Int. Conf. on Web Information Systems (WEBIST 2008)*, Lisabo, Portugal, May 2008, 190-196.
- [8] Y. Chen, Tree Embedding and XML Query Evaluation, *Int. Conf. on Enterprise Information Systems (ICEIS-2008)*, IEEE, Funchal-madeira, Barcelona, Spain, June 12-16, 2008, pp. 173-178.
- [9] Y. Chen, Document Tree Reconstruction and Fast Twig Pattern Matching, in *Proc: International Conf. on Information and Knowledge Engineering (IKE'09)*, Monte Carlo Resort, Las Vegas, USA, July 13-16, 2009, pp. 393-399.
- [10] Y. Chen, Unordered Tree Matching and Tree Pattern Queries in XML Databases, in *Proc: 14th International Conf. on Software and Data Technology (ICSOFT'09)*, Sofia, Bulgaria, July 26-29, 2009, pp. 191-198.
- [11] Yangjun Chen: A time optimal algorithm for evaluating tree pattern queries. *SAC 2010*: 1638-1642.
- [12] B. Choi. What are real DTDs like. In *WebDB*, 2002.
- [13] J. Clark and S. DeRose. XML path language (XPath). W3C Working Draft, Nov. 1999.
- [14] D. DeHaan, D. Toman, M. Consens, and T. Ozsu. Comprehensive XQuery to SQL translation using dynamic interval encoding. In *SIGMOD*, 2003.
- [15] M. Fernandez and D. Suciu. Optimizing regular path expression using graph schemas. In *ICDE*, 1998.
- [16] M. F. Fernandez, A. Morishima, and D. Suciu. Efficient evaluation of XML middleware queries. In *SIGMOD*, 2001.
- [17] D. Florescu and D. Kossmann. Storing and querying XML data using an RDMBS. *IEEE Data Eng. Bull.*, 22(3), 1999.
- [18] IBM. DB2 XML Extender. <http://www-3.ibm.com/software/data/db2/extended/xmlext/index.html>.
- [19] S. Jain, R. Mahajan, and D. Suciu. Translating XSLT programs to efficient SQL queries. In *WWW*, 2002.
- [20] H. Kaplan, T. Milo, and R. Shabo. A comparison of labeling schemes for ancestor queries. In *SODA*, 2002.
- [21] R. Krishnamurthy, V. Chakaravarthy, R. Kaushik, and J. Naughton. Recursive XML schemas, recursive XML queries, and relational storage: XML-to-SQL query translation. In *ICDE*, 2004.
- [22] R. Krishnamurthy, R. Kaushik, and J. Naughton. XML-SQL query translation literature: The state of the art and open problems. In *Xsym*, 2003.
- [23] R. Krishnamurthy, R. Kaushik, and J. Naughton. Efficient XML-to-SQL query translation: Where to add the intelligence. In *VLDB*, 2004.
- [24] Q. Li and B. Moon. Indexing and querying xml data for regular path expressions. In *VLDB*, 2001.
- [25] M. Marx. XPath with conditional axis relations. In *EDBT*, 2004.
- [26] Microsoft SQLXML and XML mapping technologies. <http://msdn.microsoft.com/sqlxml/default.asp>.
- [27] M. Nunn. An overview of SQL server 2005 for the database developer, 2004. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnsql90/html/sql_ovyukondrev.asp.
- [28] Oracle. Oracle9i XML Database Developer's Guide – Oracle XML DB Release 2. <http://otn.oracle.com/tech/xmldb/content.html>.
- [29] P. Roy, S. Seshadri, S. Sudarshan, and S. Bhobe. Efficient algorithms for multi query optimization. In *SIGMOD*, 2000.
- [30] J. Shanmugasundaram, J. Kiernan, E. J. Shekita, C. Fan, and J. Funderburk. Querying XML views of relational data. In *VLDB*, 2001.
- [31] J. Shanmugasundaram, K. Tufte, G. He, C. Zhang, D. De-Witt, and J. Naughton. Relational databases for querying XML documents: Limitations and opportunities. In *VLDB*, 1999.

- [32] W. Fan, J.X. Yu, J. Li, B. Ding and L. Qin, Query Translation from XPath to SQL in the Presence of Recursive DTDs. *The VLDB Journal* (2009) 18:857-883.
- [33] R. Elmasri and S.B. Navathe, *Fundamentals of Database Systems*, 3rd edition, Addison-Wesley, 5th edition, 2007.

APPENDIX

In the Appendix, we describe Fan's algorithm [25] and apply it to a simple graph to see how it works. Especially, showing that even for a simple graph the created regular expression can be very large.

In Fan's algorithm, the nodes in a graph are numbered, and a variable $M[i, j, k]$ is used to store the expression representing all paths from node i to node j via nodes whose numbers are less than or equal to k .

Through a nested loop, the algorithm checks all possible values for i , j , and k ; and for each combination the value of the corresponding $M[i, j, k]$ is established. Therefore, it is a brute-force algorithm.

Algorithm CycleE(G, A, B)

Input: a graph G with n nodes, and two nodes A and B in G . Output: a regular expression representing all paths from A to B in G .

begin

```

1.   for  $i = 1$  to  $n$  do {
2.     for  $j = 1$  to  $n$  do {
3.       if  $i = j$ 
4.         then  $M[i, j, 0] \leftarrow \phi$ 
5.       else if  $i \neq j$  and  $(i, j)$  is an edge  $e$  in  $G$ 
6.         then  $M[i, j, 0] \leftarrow e$ ;
7.       else  $M[i, j, 0] \leftarrow \phi$ . }
8.     for  $k = 1$  to  $n$  do {
9.       for  $j = 1$  to  $n$  do {
10.        for  $j = 1$  to  $n$  do {
11.          if  $M[i, k, k - 1] \neq \phi$  and  $M[k, j, k - 1] \neq \phi$ 
12.          then  $M[i, j, k] \leftarrow M[i, j, k - 1] \cup$ 
                $M[i, k, k - 1] \cdot M[k, k, k - 1]^* \cdot M[k, j, k - 1];$ 
13.        else  $M[i, j, k] \leftarrow M[i, j, k - 1]; } } }
14.   return  $M[A, B, n]$ ;
end$ 
```

In the algorithm, all $M[i, j, 0]$'s are first initialized (lines 1 – 7). Then $M[i, j, k]$'s with $k \geq 1$ are calculated, by inspecting $M[i, j, k - 1]$, $M[i, k, k - 1]$, and $M[k, j, k - 1]$, including all possible cycles, i.e., $M[k, k, k - 1]^*$ (lines 8 – 13).

The following example helps for illustration.

Example 5 In this example, we apply the algorithm to the graph shown in Fig. 9, and trace the computation process.

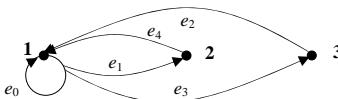


Fig. 9. A directed graph

$$\begin{aligned} k &= 1, i = 1, j = 1, \\ M[i, k, k-1] &= M[1, 1, 0] = e_0 \end{aligned}$$

$$\begin{aligned} M[k, j, k-1] &= M[1, 1, 0] = e_0 \\ M[i, j, k] &= M[i, j, k-1] \cup (M[i, k, k-1] \cdot M[k, k, k-1]^* \cdot M[k, j, k-1]) \\ M[1, 1, 1] &= M[1, 1, 0] \cup (M[1, 1, 0] \cdot M[1, 1, 0]^* \cdot M[1, 1, 0]) \\ &= e_0 \cup (e_0 \cdot e_0^* \cdot e_0) = e_0 \cup e_0^*. \\ k &= 1, i = 1, j = 2. \\ M[i, k, k-1] &= M[1, 1, 0] = e_0 \\ M[k, j, k-1] &= M[1, 2, 0] = e_1 \\ M[i, j, k] &= M[i, j, k-1] \cup (M[i, k, k-1] \cdot M[k, k, k-1]^* \cdot M[k, j, k-1]) \\ M[1, 2, 1] &= M[1, 2, 0] \cup (M[1, 1, 0] \cdot M[1, 1, 0]^* \cdot M[1, 2, 0]) \\ &= e_1 \cup (e_0 \cdot e_0^* \cdot e_1) = e_1 \cup e_0^* \cdot e_1. \\ k &= 1, i = 1, j = 3. \\ M[i, k, k-1] &= M[1, 1, 0] = e_0 \\ M[k, j, k-1] &= M[1, 3, 0] = e_3 \\ M[i, j, k] &= M[i, j, k-1] \cup (M[i, k, k-1] \cdot M[k, k, k-1]^* \cdot M[k, j, k-1]) \\ M[1, 3, 1] &= M[1, 3, 0] \cup (M[1, 2, 0] \cdot M[1, 1, 0]^* \cdot M[1, 3, 0]) \\ &= \phi \cup (e_3 \cdot e_0^* \cdot e_1) = e_3 \cdot e_0^* \cdot e_1. \\ k &= 1, i = 2, j = 1. \\ M[i, k, k-1] &= M[2, 1, 0] = e_4 \\ M[k, j, k-1] &= M[1, 2, 0] = e_1 \\ M[i, j, k] &= M[i, j, k-1] \cup (M[i, k, k-1] \cdot M[k, k, k-1]^* \cdot M[k, j, k-1]) \\ M[2, 1, 1] &= M[2, 1, 0] \cup (M[2, 1, 0] \cdot M[1, 1, 0]^* \cdot M[2, 1, 0]) \\ &= e_4 \cup (e_4 \cdot e_0^* \cdot e_0) = e_4 \cup e_4 \cdot e_0^*. \\ k &= 1, i = 2, j = 2. \\ M[i, k, k-1] &= M[2, 1, 0] = e_4 \\ M[k, j, k-1] &= M[2, 2, 0] = e_2 \\ M[i, j, k] &= M[i, j, k-1] \cup (M[i, k, k-1] \cdot M[k, k, k-1]^* \cdot M[k, j, k-1]) \\ M[2, 2, 1] &= M[2, 2, 0] \cup (M[2, 1, 0] \cdot M[1, 1, 0]^* \cdot M[2, 2, 0]) \\ &= \phi \cup (e_2 \cdot e_0^* \cdot e_1) = e_2 \cdot e_0^* \cdot e_1. \\ k &= 1, i = 2, j = 3. \\ M[i, k, k-1] &= M[2, 1, 0] = e_4 \\ M[k, j, k-1] &= M[1, 3, 0] = e_3 \\ M[i, j, k] &= M[i, j, k-1] \cup (M[i, k, k-1] \cdot M[k, k, k-1]^* \cdot M[k, j, k-1]) \\ M[1, 3, 1] &= M[1, 3, 0] \cup (M[1, 2, 0] \cdot M[1, 1, 0]^* \cdot M[1, 3, 0]) \\ &= e_4 \cup (e_4 \cdot e_0^* \cdot e_0) = e_4 \cup e_4 \cdot e_0^*. \\ k &= 1, i = 3, j = 2. \\ M[i, k, k-1] &= M[3, 1, 0] = e_2 \\ M[k, j, k-1] &= M[2, 2, 0] = e_1 \\ M[i, j, k] &= M[i, j, k-1] \cup (M[i, k, k-1] \cdot M[k, k, k-1]^* \cdot M[k, j, k-1]) \\ M[3, 2, 1] &= M[3, 2, 0] \cup (M[3, 1, 0] \cdot M[1, 1, 0]^* \cdot M[3, 2, 0]) \\ &= \phi \cup (e_2 \cdot e_0^* \cdot e_1) = e_2 \cdot e_0^* \cdot e_1. \\ k &= 1, i = 3, j = 3. \\ M[i, k, k-1] &= M[3, 1, 0] = e_2 \\ M[k, j, k-1] &= M[1, 3, 0] = e_3 \\ M[i, j, k] &= M[i, j, k-1] \cup (M[i, k, k-1] \cdot M[k, k, k-1]^* \cdot M[k, j, k-1]) \\ M[3, 3, 1] &= M[3, 3, 0] \cup (M[3, 1, 0] \cdot M[1, 1, 0]^* \cdot M[3, 3, 0]) \\ &= \phi \cup (e_2 \cdot e_0^* \cdot e_3) = e_2 \cdot e_0^* \cdot e_3. \\ k &= 2, i = 1, j = 1. \\ M[i, k, k-1] &= M[1, 2, 1] = e_1 \cup e_0^* \cdot e_1 \\ M[k, j, k-1] &= M[2, 1, 1] = e_1 \cup e_4 \cdot e_0^* \\ M[i, j, k] &= M[i, j, k-1] \cup (M[i, k, k-1] \cdot M[k, k, k-1]^* \cdot M[k, j, k-1]) \\ M[1, 1, 2] &= M[1, 1, 1] \cup (M[1, 2, 1] \cdot M[2, 2, 1]^* \cdot M[1, 2, 1]) \\ &= e_0 \cup e_0^* \cup ((e_1 \cup e_0^* \cdot e_1) \cdot (e_4 \cdot e_0^* \cdot e_1)^* \cdot (e_4 \cup e_4 \cdot e_0^*)). \square \end{aligned}$$

From the sample trace, we can see that the expressions produced by Fan's algorithm tend to be large. For instance, the expression representing all paths from node 1 to node 1 is

$$e_0 \cup e_0^* \cup ((e_1 \cup e_0^* \cdot e_1) \cdot (e_4 \cdot e_0^* \cdot e_1)^* \cdot (e_4 \cup e_4 \cdot e_0^*)).$$

But the minimized regular expression for this is $(e_0 \cup e_4 \cup e_2)^*$, which can be obtained by doing a computation similar to Example 1 since the graph is reducible.

Large Software Component Repositories into Small Index Files

Marcos Paulo Paixão, Leila Silva

Computation Department
Federal University of Sergipe
Aracaju, Brazil

marcospsp@decomp.ufs.br, leila@ufs.br

Talles Brito, Gledson Elias

Informatics Department
Federal University of Paraíba
João Pessoa, Brazil

talles@compose.ufpb.br, gledson@di.ufpb.br

Abstract—Software component repositories have adopted semi-structured data models for representing syntactic and semantic features of handled assets. Such models imply challenges to search engines, which are related to the design of indexing techniques that ought to be efficient in terms of storage space requirements. In such a context, by applying clustering techniques before indexing component repositories, this paper proposes an approach for reducing the number of assets in the repository, and consequently, the size of index files. Based on an illustrative repository, outcomes indicate a significant optimization in the number of assets to be indexed.

Keywords - *Component repositories; indexing; clustering techniques.*

I. INTRODUCTION

By enabling different software developers to share software assets, software component repositories have the potential to improve software reuse level. However, reuse of software assets is in general a hard task, particularly when search and selection must be conducted over large-scale asset collections. Therefore, in repository systems, it is important the development of search engines that can help searching, selecting and retrieving required software assets.

According to Orso *et al.* [11], the aim of a repository system is not to store software assets only, but also metadata describing them. Such metadata provides information employed by search engines for indexing stored assets. In such a direction, as endorsed by Vitharana [13], component description models can adopt high level concepts for describing component metadata, making possible to express syntactic and semantic features, and so, facilitating developers to search, select and retrieve assets. In practice, currently available component description models have adopted approaches based on semi-structured data, more specifically XML, allowing structural relationships among elements to aggregate semantic to textual values. As examples, it can be mentioned RAS [10] and X-ARM [3].

However, indexing techniques based on textual restrictions are not efficient for semi-structured data. Such techniques are unable of indexing structural relationships among terms, compromising query precision with false-positives. Thus, the adoption of semi-structured data implies challenges related to the design of indexing techniques that ought to be efficient in terms of storage space requirements, processing time and precision level of queries, which can be constrained by textual and structural restrictions.

Several proposals can be found in the literature for dealing with such problems. Despite their relevant contributions, existing techniques do not meet storage space and query processing time requirements [9], and also query precision level [6]. In such a scenario, the proposal presented by Brito *et al.* [1] represents a noticeable indexing technique based on semi-structured data, which can be considered precise and efficient in terms of query processing time, but suffer from problems related to storage space requirements. Such problems occur because generated index files are bigger than the input database. Thus, in the context of large-scale software component repositories, it is still a challenging open issue to design indexing techniques that minimize the storage space requirements without excessively impacting on query processing time and precision.

In such a context, based on the adoption of clustering techniques, this paper proposes an approach for reducing the number of assets in the repository, and consequently, optimizing the storage space requirements. Taking into account a large-scale component repository, the proposed approach identifies clusters (groups) of similar software assets and generates new representative assets, which in turn must be handled by the indexing technique supported by the search engine of the repository. Each representative asset has a simplified description, also based on semi-structured data, which makes reference to all original assets that belong to its cluster of similar assets. In order to do that, the paper also proposes a similarity metric that has the aim of indicating the set of assets that belongs to the same cluster. The bigger the similarity among assets in the repository, the lesser is the number of identified clusters, and as a result, the lesser is the number of representative assets that must be indexed by the search engine, enabling to save storage space.

The remainder of this paper is structured as follows. Section II describes related techniques, evincing the original initiative of applying clustering techniques in the context of indexing software component repositories. The adopted component description model, called X-ARM, is briefly presented in Section III, identifying the main types of assets and their relationships. Then, Section IV presents the proposed clustering approach for reducing the number of assets to be indexed, and so, optimizing storage space requirements. After that, some outcomes observed in a preliminary evaluation performance are presented in Section V. In conclusion, Section VI presents some final remarks and delineates future work.

II. RELATED TECHNIQUES

Taking into account that the problem of data clustering is NP-hard, several heuristics have already been proposed. Xu and Wunsch [15] present an interesting review of the research field. In [4], Feng shows that clustering algorithms, in particular, hierarchical algorithms and K-Means [7], are equivalent to optimization algorithms of a fitness function.

In this paper, a two-stage, heuristic clustering approach is proposed, based on the classical hierarchical algorithm and K-Means. In order to validate the proposed approach, a random database composed of 27.000 assets has been generated and results indicate that there is a significant optimization in terms of the number of assets to be indexed.

However, for the best knowledge of the authors, clustering techniques have never been adopted in the context of indexing software components repositories. Therefore, it seems an original contribution to apply such techniques when indexing component repositories. Despite the mentioned originality, several other proposals have already adopted clustering techniques in problems of the software engineering. For instance, Wu *et al.* [14] compares several clustering approaches proposed in the context of software evolution. In [8], Li *et al.* proposes the adoption of clustering techniques for encapsulating software requirements. Chiricota *et al.* [2] investigates the application of clustering techniques in the domain of reverse engineering, in particular, adopting such techniques to recover the structure of software systems.

III. X-ARM

In order to express syntactic and semantic features of software components, Frakes [5] suggests the adoption of component description models, which provide a set of information that allows search systems to index and classify all types of related assets. In such a direction, this paper explores the X-ARM description model, which adopts a XML-based semi-structured data model, expressing not only syntactic information but also semantic properties [3]. Besides, X-ARM enables describing several types of software assets, which can be produced in component-based development processes, proving the required semantic for representing their relationships.

As illustrated in Fig. 1, X-ARM allows describing component and interface specifications, as well as component implementations. The component and interface specifications can be described in a way that is independent or dependent of component model. On the one hand, independent specifications do not take into account any feature or property of component models, such as CCM, JavaBeans, EJB and Web Services. On the other hand, dependent specifications ought to consider features and properties related to the adopted component models.

In X-ARM, both dependent and independent interface specifications are described as a set of operations. Each operation has a name, a set of input or output parameters and a return value. In component-based development processes, dependent interface specifications must be in conformance with their independent counterparts. So, in Fig. 1, it can be

observed that dependent interface specifications must reference to their respective independent interface specifications.

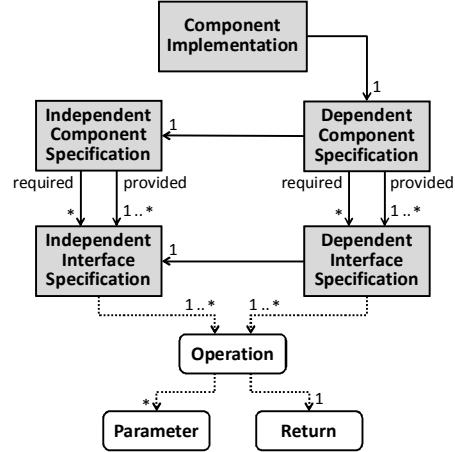


Figure 1. Relationships between artifacts.

Dependent and independent component specifications can make reference to a set of provided and required interface specifications. However, it must be noticed that independent component specifications can refer to independent interface specifications only. Similarly, dependent component specifications can refer to dependent interface specifications only. In component-based development processes, dependent component specifications must be in conformance with their respective independent counterparts. Therefore, note that dependent component specifications must make reference to their respective independent component specifications.

In summary, dependent interface and component specifications must be in conformance with their respective independent specifications. Besides, for each independent specification, several dependent specifications can be described, each one in conformance with a given software component model.

In a similar way, in component-based development processes, component implementations must be in conformance with their respective dependent component specifications. So, in Fig. 1, note that component implementations must refer to their correspondent dependent component specifications. Besides, for each dependent component specification, several component implementations can be realized.

As an example of the description of an asset in X-ARM, Fig. 2 illustrates a fragment of a dependent component specification. In Fig. 2, all lines are numbered and many details have been suppressed for didactic purposes. Line 1 represents the asset header, in which can be found the asset identifier (id). Lines 2 to 4 make reference to the independent component specification, from which the described asset must be in conformance with. Then, Lines 5 to 14 refer to all dependent interface specifications, which are provided by the described dependent component specification. Although note illustrated in Fig. 2, required interfaces can also be specified in a similar way.

```

01 <asset name="dependentCompSpec-X"
        id="compose.dependentCompSpec-X-1.0-beta">
02   <model-dependency>
03     <related-asset name="independentCompSpec-Z"
                       id="compose.independentCompSpec-Z-1.0-stable"
                       relationship-type="independentComponentSpec"/>
04   </model-dependency>
05   <component-specification>
06     <interface>
07       <provided>
08         <related-asset name="dependentInterface-A"
                           id="compose.dependentIntSpec-A-2.0-stable"
                           relationship-type="dependentInterfaceSpec"/>
09       </provided>
10      <provided>
11        <related-asset name="dependentInterface-B"
                           id="compose.dependentIntSpec-B-3.0-stable"
                           relationship-type="dependentInterfaceSpec"/>
12      </provided>
13    </interface>
14  </component-specification>
15 </asset>

```

Figura 2. Component specification in X-ARM.

IV. A CLUSTERING BASED INDEXING APPROACH

As largely recognized in the literature, the task of indexing repositories based on semi-structured data is a relevant issue [1][6][9]. One of the major challenges is to provide an indexing mechanism that reduces storage space requirements, but without excessively impacting on query processing time and precision level.

In such a context, this paper proposes a solution for optimizing the storage space required by index files. To do that, the proposed approach constructs a clustered repository, which is composed of representative assets of the set of software assets stored in the original repository. Therefore, instead of indexing the original repository, the adopted search service ought to index the reduced set of representative assets, which make reference to the original assets. In order to identify the groups of similar assets, and, consequently, to construct the representative assets that compose each group, the paper also proposes the adoption of data clustering techniques.

Clustering techniques [7] consist of three basic phases: (i) extraction of features that express the behavior of the elements to be clustered; (ii) definition of the similarity metric in order to compare evaluated elements; and (iii) adoption of a clustering algorithm. The phase of extracting features consists in defining what information is relevant to express the evaluated element and how information is quantified. Such information defines an attribute vector and thus an element can be represented as a point in the multidimensional space. The similarity metric expresses in quantitative terms the similarity between elements. In general, a function is defined for such a purpose, in which the Euclidean distance [7] between two points (elements) is one of the more common adopted metrics. Finally, the data clustering algorithm is a heuristic that has the aim of generating groups of elements, in which each group is composed of similar elements, according to the adopted similarity metric.

A. Relevant Features

The approach proposed herein applies the clustering technique taking into account the five types of assets that can be stored in the repository, that is: dependent and independent component specifications, dependent and independent interface specifications and component implementations. The clustering technique is applied separately for each type of asset. Therefore, each type has a distinct attribute vector for representing its features.

For each component implementation, the relevant feature is its referenced dependent component specification. Hence, different implementations of the same dependent component specification are considered similar.

In turn, for each dependent component specification, the relevant features are its referenced independent component specification as well as its set of provided dependent interface specifications. Therefore, different dependent component specifications are considered similar when they refer to the same independent component specification or have in common a considerable subset of provided dependent interface specifications.

In relation to independent component implementations, the relevant feature of each one is its set of provided independent interface specifications. So, different independent component specifications are considered similar when they have in common a considerable subset of provided independent interface specifications.

Taking into account dependent interface specifications, the relevant features of each one are its referenced independent interface specification together with their operations. Thus, different dependent interface specifications are considered similar when they refer to the same independent interface specification or have in common a considerable subset of defined operations.

Finally, for each independent interface specification, the relevant features are its defined operations, considering their names, input and output parameters and the return value. Consequently, different independent interface specifications are considered similar when they have in common a considerable subset of defined operations.

As an example, Table I presents the attribute vector of the asset illustrated before in Fig. 2. As can be noticed, the asset is a dependent component specification. Therefore, the attribute vector is composed of its referenced independent component specification (lines 2 to 4) and its set of provided dependent interface specifications (lines 5 to 14).

TABLE I. ATTRIBUTE VECTOR OF THE ASSET X.

ID	compose.dependentCompSpec-X-1.0-beta
Independent Component Specification	compose.independentCompSpec-Z-1.0-stable
Dependent Interface Specification	compose.dependentIntSpec.A-2.0-stable compose.dependentIntSpec.B-3.0-stable

B. Similarity Metric

The similarity metric is defined based on the attribute vector of the asset. Since the attribute vector differs between distinct types of assets, the similarity metric is also different for each type of asset. Due to space limitation, the adopted metrics are not completely described (see [12] for details). In order to illustrate the composition of the metric, consider the case of determining the similarity between two dependent component specifications. In such a case, if two dependent component specifications have the same reference to a given independent component specification, then a certain value, called distance, is assigned to the similarity among them. Besides, the intersection and union sets of their provided dependent interface specifications are calculated. A weight is assigned to the ration among the size of the intersection and union sets in such a way that dependent component specifications are considered more similar when the ration is closer to one, and considered more different when the ration tends to zero.

As an example of calculating the similarity metric, consider the dependent component specifications that have the attribute vectors illustrated in Table I and II. The similarity between them is established using their attribute vectors. Such a similarity is expressed by a numeric value, which can be calculated according the following equation:

$$D_f = D_i - k - (\text{intersection}/\text{union}) * 100. \quad (1)$$

In Eq. (1), the terms have the following values. The term D_i is a default initial distance ($D_i = 300$). In turn, the term k can be the value 200, if both specifications make reference to the same dependent component specification, or otherwise the value 0. The term *intersection* expresses the number of provided dependent interfaces that both specifications have in common. Finally, the term *union* represents the number of provided dependent interfaces that both specifications have together. In the example, $D_i = 300$, $k = 0$; *intersection* = 1 and *union* = 3. Thus, $D_f = 300 - 0 - 33 = 267$.

TABLE II. ATTRIBUTE VECTOR OF THE ASSET Y.

ID	compose.dependentCompSpec-Y-1.0-beta
Independent Component Specification	compose.independentCompSpec-W-1.0-beta
Dependent Interface Specification	compose.dependentIntSpec.A-2.0-stable compose.dependentIntSpec.C-1.0-stable

C. Clustering Algorithm

The proposed clustering algorithm has two stages. In the first stage, the classical hierarchical clustering algorithm [7] is applied adopting the concept of threshold. Thus, the clustering algorithm is performed until the distance between the clusters is greater than the threshold, which is specified by the user. For each identified cluster, a representative asset is constructed and stored in nonvolatile memory. In order to make the performance better, the implementation of the

algorithm loads the assets to be clustered in volatile memory, until reaching its storage capacity. During such a stage, the assets are randomly selected from the repository.

Fig. 3 illustrates the main steps of the first stage: (a) assets are randomly selected from the repository; (b) clusters composed of similar assets are constructed by applying the hierarchical clustering algorithm; and (c) representative assets are created for representing each cluster.

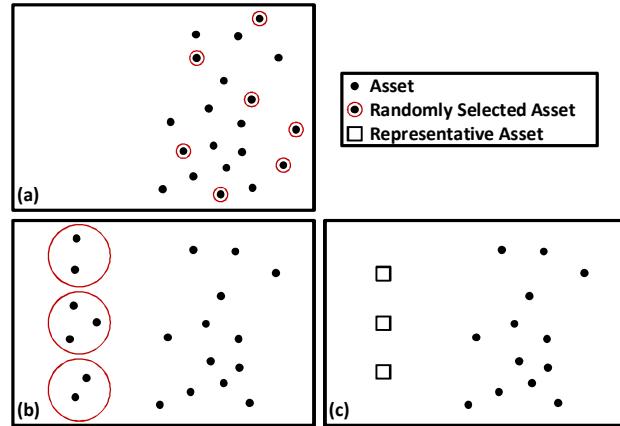


Figura 3. The first stage.

In the second stage, a K-Means based algorithm [7] is adopted. In general terms, representative elements are considered centroids. However, differently from K-Means, such centroids are not recalculated in the proposed approach. Indeed, each asset, not yet clustered in the first stage, is compared with each representative asset. The asset is candidate to be included in a cluster when the distance between the asset and the respective representative asset is lesser than the threshold. Fig. 4 shows the second stage.

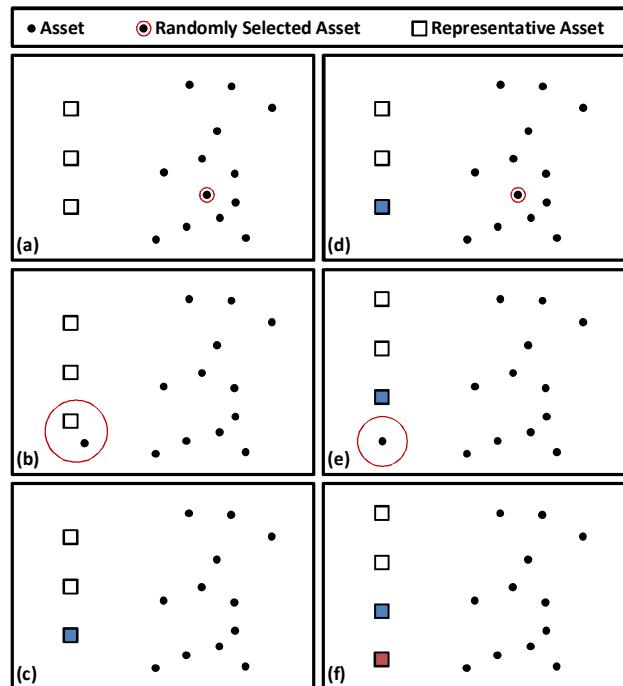


Figure 4. The second stage.

As depicted in Figs. 4a, 4b, and 4c, considering all candidate clusters, the asset is included in the cluster that has the minor distance and then the representative element of the cluster is reconstructed considering the features of the included asset. Otherwise, as shown in Figs. 4d, 4e and 4f, if the asset is not a candidate to any cluster, the own asset becomes a new representative element and so a new cluster.

V. PERFORMANCE EVALUATION

In order to evaluate the proposed approach, it has been developed a customizable script that automatically generates a repository that stores the mentioned X-ARM assets. The generated repository has 27.000 different types of assets. After creating the repository, the proposed approach has been applied for grouping the stored assets in clusters, generating their respective representative assets. Fig. 5 presents the number of each type of asset in the original repository and the clustered repositories after the application of the proposed approach using the threshold of 175 and 150.

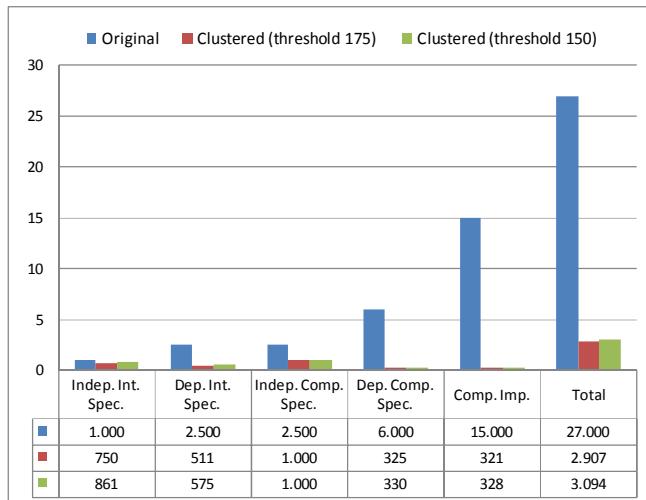


Figure 5. Number of Assets.

As can be noticed in Fig. 5 and Fig. 6, the proposed approach significantly reduces the original repository. For example, when the threshold is 175, the number of stored assets in the original repository is reduced around 89,2%, dropping from 27.000 original assets to 2.907 representative assets. In terms of storage space, the proposed approach reduces the storage space requirements around 43%, dropping from 18 MB in the original repository to 10 MB in the clustered one.

When the threshold is reduced, as expected, more representative elements can be constructed because more clusters are created. Thus, when the threshold is reduced from 175 to 150, the number of original assets is reduced from 27.000 to 3.094 representative assets, which still represents a significant reduction in the number of stored assets around 88,5%.

Consequently, as illustrated in Fig. 6, in terms of the number of assets, the gains of applying the proposed approach are significantly relevant, varying between 89,2% and 88,5% for the thresholds of 175 and 150, respectively.

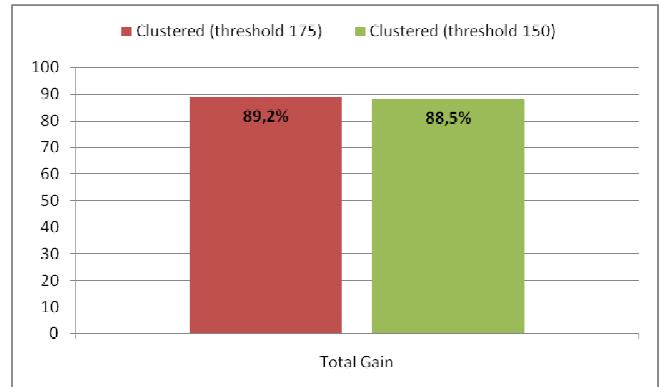


Figure 6. The total gain

However, as depicted in Fig. 7, the gains are different for each type of asset. For independent interface specifications, the gains are around 25% and 13,9% for thresholds of 175 and 150, respectively. Such lower gains can be explained by the difficulty of finding two or more interfaces that has a reasonable set of common operations, which are evaluated in terms of their names, the types of their input and output attributes, and also the type of their return values.

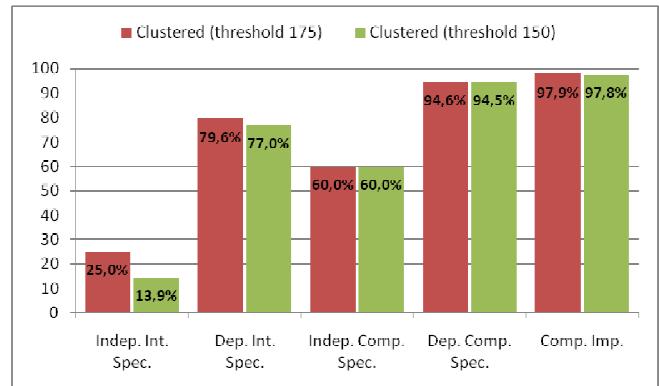


Figure 7. The gains for different types of assets

Considering dependent interface specifications, the gains become more expressive, increasing to 79,6% and 77% when thresholds are 175 and 150, respectively. Part of the reason for that is that, during the generation of the original repository, the adopted script creates 2 or 3 dependent interfaces that refer to the same independent interface, representing that each independent interface is specified for at least 2 or 3 different component models in practice. So, as the similarity metric for dependent interfaces is based on their referenced independent interface together with their operations, it is already expected such expressive gains, as demonstrated in the experiments.

In relation to independent component specifications, the gains are around 60% for both thresholds. Such gains are relatively high and indeed not expected. However, as mentioned before, independent component specifications are considered similar when they have in common a considerable subset of provided independent interfaces. Considering that independent interfaces have expressive clustering rates, such gains make possible to group several

interfaces in a unique representative interface, increasing the likelihood of independent component specifications to refer to the same provided interfaces, and consequently, justifying the high gains for both thresholds.

In terms of dependent component specifications, the gains become much more expressive, increasing to 94,6% and 94,5% when thresholds are 175 and 150, respectively. The rationale for that is that, during the generation of the original repository, the adopted script creates 2 or 3 dependent components that refer to the same independent component, representing that each independent component is specified for at least 2 or 3 different component models in practice. So, such better gains are understandable because the similarity metric for dependent components is based on their referenced independent components, which already have expressive clustering rates.

Finally, for component implementations, the gains become higher, around 97,9% and 97,8% when thresholds are 175 and 150, respectively. Again, the rationale for that is that, when generating the original repository, the adopted script creates 2 or 3 component implementations for each dependent component specification, representing that each dependent component has at least 2 or 3 different implementations in practice. So, such higher gains are expected because the similarity metric for component implementations is based on their referenced dependent components, which already have expressive clustering rates.

As can be noticed, the clustering gains in independent interfaces specifications impact on the gains in both dependent interfaces specifications and independent component specifications. Similarly, the clustering gains in independent component specifications impact on the gains in dependent component specifications, which in turn impact on the gains in component implementations.

VI. CONCLUSION

Based on the preliminary results, it can be clearly evinced as benefits the potential of the proposed approach in significantly clustering an X-ARM repository and consequently reducing storage space requirements. It must be highlighted that, the bigger the original repository in terms of the number of stored assets, the more expressive the likelihood of clustering assets, and so the better the gain in terms of storage space requirements.

Taking into account that the indexing technique proposed by Brito *et al.* [1] will be adopted for indexing the clustered repository, it is taken for granted that the reduction in the size of the original repository implies in an expressive reduction in the size of index files of the clustered repository. Besides, considering that the technique proposed by Brito *et al.* has an excellent performance in query processing time, even in large-scale index files, it is expected a reasonable gain in terms of query processing time due to the expressive reduction in the size of index files. Therefore, the proposed approach clearly makes possible to map large software component repositories into small index files.

However, as often informally said, there is no free lunch. That is, in formal words, such expressive gains in terms of storage space requirements and query processing time,

almost certainly have an impact on the query precision level, since the process of clustering assets introduces some degree of information loss in representative assets. It must be stressed that the tradeoff between the best threshold and the query precision level has not yet been investigated. In such a sense, the evaluation of the impact of the proposed approach in terms of query processing time and precision level constitutes future work. Besides, it is also under investigation a comparative analysis contrasting the proposed approach and other ones available in the literature, but applied in different research fields.

ACKNOWLEDGMENT

This work was supported by the National Institute of Science and Technology for Software Engineering (INES – www.ines.org.br), funded by CNPq, grants 573964/2008-4.

REFERENCES

- [1] T. Brito, T. Ribeiro, and G. Elias, "Indexing Semi-Structured Data for Efficient Handling of Branching Path Expressions", 2nd Inter. Conf. on Advances in Databases, Knowledge, and Data Applications (DBKDA 2010), France, 2010, pp. 197-203.
- [2] Y. Chiricota, F. Jourdan, and G. Melançon, "Software Component Capture using Graph Clustering", Proc. IEEE International Workshop on Program Comprehension, 2003.
- [3] G. Elias, M. Schuenck, Y. Negócio, J. Dias, and S. Miranda, "X-ARM: An Asset Representation Model for Component Repository", Proc. 21st ACM Symposium on Applied Computing (SAC 2006), France, 2006, pp. 1690-1694.
- [4] A. Feng, "Document Clustering – An Optimization Problem", ACM SIGIR 2007, pp. 819-820.
- [5] W. Frakes and K. Kang, "Software Reuse Research: Status and Future", IEEE Transactions on Software Engineering, vol.31, issue 7, July 2005, pp. 529-536.
- [6] R. Goldman and J. Widom, "DataGuides: Enabling Query Formulation and Optimization in Semistructured Databases", Proc. 23rd Int. Conf. on Very Large Data Bases (VLDB 1997), Greece, 1997, pp. 436-445.
- [7] A.K. Jain and R.C. Dubes, Algorithms for Clustering Data, Prentice Hall, 1984.
- [8] Z. Li, Q.A. Rahman, and N.H. Madhavji, "An Approach to Requirements Encapsulation with Clustering", Proc. 10th Workshop on Requirement Engineering, 2007, pp. 92-96.
- [9] W. Meier, "eXist: An Open Source Native XML Database", NODE 2002 Web and Database-Related Workshops on Web, Web-Services, and Database Systems, 2002.
- [10] OMG, Reusable Asset Specification: OMG Available Specification – v2.2, 2005.
- [11] A. Orso, M.J. Harrold, and D.S. Rosenblum, "Component Metadata for Software Engineering Tasks", Proc. 2nd Int. Workshop on Engineering Distributed Objects, 2000, pp. 126-140.
- [12] M.P.S. Paixão, T.B. Viana, L. Silva, and G. Elias, G. "Optimizing the Search Space in Distributed Component Repositories", Technical Report, June 2010. <http://www.compose.ufpb.br/reports/component-repository-cluster.pdf> (in Portuguese).
- [13] P. Vitharana, F. Zahedi, and H. Jain, "Knowledge-Based Repository Scheme for Storing and Retrieving Business Components: A Theoretical Design and an Empirical Analysis", IEEE Transactions on Software Engineering., vol. 29, issue 7, July 2003, pp. 649-664.
- [14] J. Wu, A.E. Hassan, and R.C. Holt, "Comparison of Clustering Algorithms in the Context of Software Evolution", Proc. 21st Int. Conf. on Software Maintenance, 2005, pp. 525-535.
- [15] R. Xu and D. Wunsch, "Survey of Clustering Algorithms", IEEE Transactions on Networks, vol.16, issue 3, May, pp. 645-678.

Intelligent Database Flexible Querying System by Approximate Query Processing

Oussama Tlili

Faculty of Sciences of Tunis
Campus Universitaire, 1060 Tunis, Tunisia
tlili.oussama@gmail.com

Minyar Sassi

National Engineering School of Tunis
BP. 37, Le Belvédère, 1002 Tunis, Tunisia
minyar.sassi@enit.rnu.tn

Habib Ounelli

Faculty of Sciences of Tunis
Campus Universitaire, 1060 Tunis, Tunisia
habib.ounelli@fst.rnu.tn

Abstract— Database flexible querying is an alternative to the classic one for users. The use of Formal Concepts Analysis (FCA) makes it possible to make approximate answers that those turned over by a classic DataBase Management System (DBMS). Some applications do not need exact answers. However, flexible querying can be expensive in response time. This time is more significant when the flexible querying require the calculation of aggregate functions (“Sum”, “Avg”, “Count”, “Var”, etc.). In this paper, we propose an approach which tries to solve this problem by using Approximate Query Processing (AQP).

Keywords - *Flexible Querying; Approximate Queries; Formal Concept Analysis; Sampling.*

I. INTRODUCTION

A flexible querying technique is used to enhance access and human interaction with information systems and to make it easier for users to find what they are looking for.

It tries to make the classic DB querying more flexible for users. To this effect, several approaches have been proposed in the literature such as additional criteria [1][2], preferences [3], distance and similarity [4][5], models based on the fuzzy-sets theory [6][7], approaches based on Type Abstraction Hierarchies (TAH) and Multi-Attributes Type Abstraction Hierarchies (MTAH) [8], and recently approaches based on the FCA [9] and those based on fuzzification of the FCA [10]. These approaches have some limits. We can mention the following:

1) No consideration of aggregate queries: they not support the aggregation functions such as *Average*, *Count*, *Max*, *Min* and *Sum*.

2) Accuracy of the answer: in many applications, the accuracy of the answer to the last decimal is not required. The user wants approached answers as soon as possible instead of waiting more time for the exact response.

3) Response time: in the case of large DB, the time taken to build the final response is enormous.

For aggregation queries, we propose a way to data route using FCA to generate a hierarchy allowing the user to personalize these responses into several levels.

For answer accuracy, we propose to use Approximate Query Processing (AQP) which consists of techniques that sacrifice accuracy to improve response time.

To improve response time, we propose to adapt the online aggregation [11] whose objective is to gradually approximate answers when running the application. It consists of applying a sample on the initial data of the DB to minimize disk access and therefore improve response time.

This paper is organized as follows. After the introduction, Section 2 presents a state of the art on flexible querying systems recently proposed and techniques of AQP. In Section 3, we propose the architecture of our system. In Section 4, we detail the various steps of the proposed approach. In Section 5, we present a general description of our approach by an illustrative example. In Section 6, we make a comparative study between the proposed approach and approaches similar to ours. In Section 7, we evaluate our approach. Finally, we summarize our work and propose future works in Section 8.

II. STATE OF THE ART

In this section, we present flexible querying systems and some AQP techniques.

A. Flexible Querying Systems

Flexible querying database try to extend the binary querying by introducing preferences in query criteria. These preferences allow for direct qualitative responses. Thus, data returned by a query will be “more or less relevant”, according to the preferences.

Research on flexible querying investigates the handling of imperfection of information (about queries), e.g., due to imprecision, uncertainty and/or incompleteness. Using traditional querying techniques, a record will only be part of the query result if it completely satisfies all the constraints imposed by the query. Due to imperfections, which often occur in reality, such an approach is too stringent. Also, in traditional querying a query is generally a complete specification of what is wanted. Flexible querying helps to relax this, making it possible that records that e.g., satisfy most (but not all) of the constraints will also be present in the query result –this is particularly useful when none of the records satisfies all constraints– and allowing query formulations to be invariably incomplete.

In this section, we limit ourselves to the approaches close to our.

Query relaxation approach proposed in [8] uses predicates with relaxing attributes. In this context, we use attributes with predicate for comparison with a linguistic term such as “Average” in place to say “Price between 200 and 300”.

This approach present two main contributions compared to others especially that of Chu *et al.* [12]. These contributions are as follows:

- Taking into account the interdependence of the search criteria query.
- Detection of inconsistencies between the search criteria before executing the query.
- Cooperation with the user by offering data near the query instead of empty answers.

However, problems of storage and indexing TAH and MTAH structures constitute a handicap to their use in the querying process.

In [10], fuzzification of the FCA in the process of flexible querying was introduced. The general principle of this approach is to organize data to optimize the query towards his given. The notion of concept application is used to allow verification of the query realisability. The returned answers were classified by satisfaction degree measured compared to the user query.

Some limits arise with this approach. We can mention: i) the response time met for answers generation, and ii) the complexity of the used structures.

A cooperative approach to flexible relational DB querying proposed in [9] based on fuzzy set theory to model the fuzzy predicates included in the query. It is based on the lattice concept to evaluate flexible queries submitted by users.

Moreover, the approach generates query causes with no answers and offers sub-queries with approximate answers.

However, the approach has several limitations such as:

- Scheduling of sub-queries approximate taking into account preferences expressed by the user in the original complaint.
- The inclusion of some widely used language modifiers like “most” and “approximately” in the query qualifiers.

All these approaches do not take into account aggregate queries and have a response time sufficiently high.

B. Approximate Queries Processing by sampling

The AQP is an effective solution which consists of techniques that sacrifice accuracy to improve response time. It is used in aggregate queries (including SUM, COUNT, AVG, etc.), whose accuracy what the “last decimal” is not required.

There are several techniques for the AQP, we can cite the sampling techniques [13], the use of histograms and Wavelets [14].

We are interested in sampling techniques. His principle is to build tables or views by selecting certain rows from the table to build an initial sample. It has a storage size smaller than the initial table, instead of questioning all the comics, the user asks a sample representing the DB and then gets an approximate answer.

The basic architecture of AQP based on sampling as described in Figure 1. It consists of two phases:

- **Offline Phase:** before executing the query, the sample is constructed from the DB tables.
- **Online Phase:** queries are rewritten to be run on the sample. The result is then measured to give the approximate response also with an error rate.

III. ARCHITECTURE OF THE PROPOSED SYSTEM

Figure 1 describes the querying flexible system architecture called FLEXTRA. We have added several components to relational DBMS such as KB (Knowledge Base).

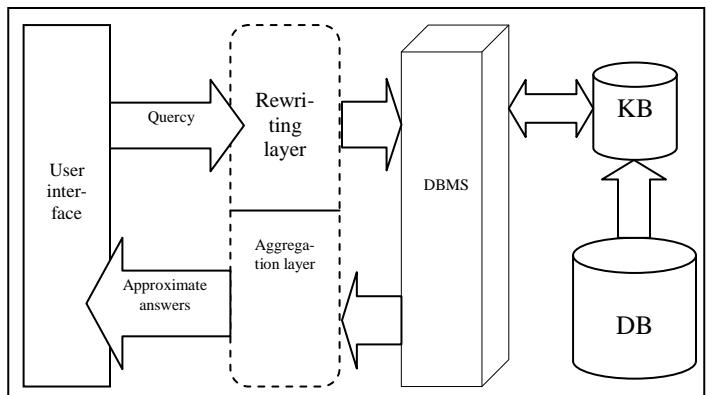


Figure 1. System Architecture

His system includes the following components:

- **Rewritable layer:** it takes care of rewriting the aggregate query in its final form by adding aggregate functions and calculating the error rate depending on the confidence degree defined by user. The query becomes an approximate query.
- **Aggregation layer:** it is responsible for transferring the user with different responses gradually during the query execution. It gives the error rate.
- **DB:** it is a relational database where we store all permanent information in a relational model.
- **KB:** it is a Knowledge Base that is generated from the DB and before the query execution. It contains information on the relaxing attributes (an attribute that describes a linguistic term). The schema is described in Table 1.

TABLE 1.KB SCHEMA

ID row	Relaxing-Attribute1	Relaxing-Attribute 2	...	Relaxing-Attribute n
...

IV. DESCRIPTION OF THE PROPOSED APPROACH

Our approach is described in Figure 2. It is divided into two major phases:

- Pre-treatment phase: in this phase we will generate the KB from the DB to contain the degrees of membership of each tuple relaxing attributes.
- Post-treatment phase: when the user launches the application, the system searches for approximate answers,

and then calculates the aggregation and gradually sends to the user.

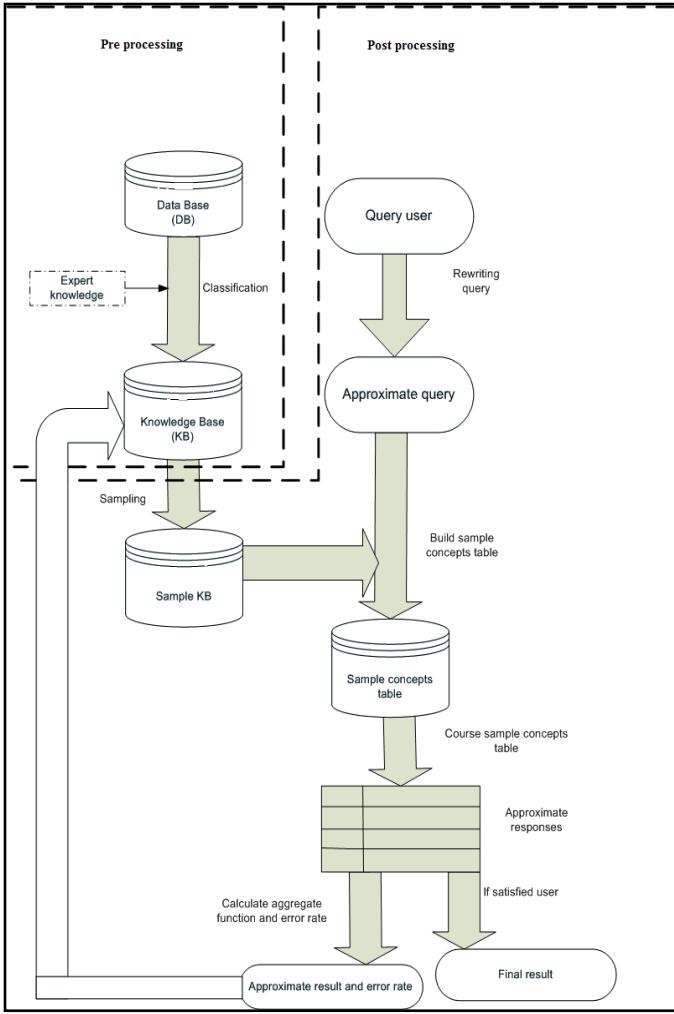


Figure 2. The approach phases

A. Building KB

Clustering allows partitioning the data into clusters, the domain expert will assign linguistic terms (e.g., young age, low salary, etc.) to use it in the query and this constitutes the KB.

A KB contains the membership degree of each tuple to relaxing attributes using the membership function. Zadeh proposes a series of fuzzy membership functions [15]; we include essentially the triangular function, the function singleton, L Function, Gamma function, and trapezoidal function.

We use a trapezoidal function, it is defined by a lower limit a , an upper limit d . Moreover, it is characterized by a lower limit b and an upper limit c to the core. This function is defined as follows:

$$u_E(x) = \begin{cases} 1 & \text{si } c \leq x \leq b \\ \frac{x-a}{b-a} & \text{si } a \leq x \leq b \\ \frac{d-x}{d-c} & \text{si } c \leq x \leq d \\ 0 & \text{si } x < a \text{ ou } x > d \end{cases}$$

Example: Table 2 presents the membership matrix on “Age” attribute; it has two reliable attributes “young” and “adult”.

TABLE 2. FUZZY CLUSTERING IN AGE ATTRIBUTE

Id row	Young Age	Adult Age
25	0.7	0.3
30	0.2	0.8
20	1	0

If Age_Young= 0.7 to 1 then the row has a membership degree = 0.7 for the Young_Age cluster.

B. Query Flexible Rewriting

The first step of query execution consists of construct the approximate query through an interface in which it specifies the confidence degree, the target table, the aggregate function (SUM, AVG, COUNT, etc.), all attributes of the SELECT clause and all attributes of the WHERE clause.

In this paper, we consider viewing a single table without using Group By knowing that it contains thousands of records. The approximate query as follows:

```
Select function(attribute), confiance_degree as confidenc,
functionInterval(confiance_degree) from table where
attributel IS flexible_condition1 [and ... attribute2 IS
flexible_condition2]
```

Where *function()* and *functionInterval()* [11] are user predefined functions and which can give online approximate answers depending on the confidence degree for aggregate AVG, SUM, COUNT, etc.

C. Sampling

The second step of our approach is to construct a sample from the KB.

The sampling is made in online mode and the gain of time is due of treatment of small KB (not all KB where construction of context table in large KB takes a long time).

Instead of querying the entire KB, we interview a sample of KB that is made up of hundreds of records which improves response time.

Administrator (expert) sets the percentage of sampling depending on the size of KB (if s : Percentage of sampling and n = the size of KB then sample size $p = (n * s)/100$).

We use the method of [11] for sampling; we randomly choose p lines from KB who have not been previously extracted.

Our approach is to build the sample using the following algorithm:

Algorithm1 : Sampling	
Inputs:	Query :Q Knowledge base :KB KB size :n Sample Percentage: s
Output:	Sample :E
Begin	
Step 1:	KB1:= KB-E.
Step 2:	E contains the $\frac{n*s}{100}$ lines extracted randomly from KB1.
Step 3:	Repeat steps 1 and 2 Until all rows have been processed.
End	

D. Building sample concepts table

The third step of our approach is to build the concepts table associated with sample building in the previous phase. The concepts table [16] is a tabular representation of a concept lattice and its construction is easier than the lattice.

The context table is a table structure but not a tree (concept lattice), and it is simple to use, modify, delete and generate concepts in the implementation step.

The context table is simply the result of a clustering operation giving membership degrees of each data to each cluster.

This is described in Table 3, where the columns have the following meanings:

- C# (context#): The name of the source context.
- Niv#, N#(Level#, Node#) :These two columns store the identifier of the concept of context. The first is the level of the concept in the lattice while the second represents the sequence number of the concept at this level.
- Int#, Ext# (Intention, Extension): These columns store for (respectively extension) of each concept.
- L_s#,L_p#(Successors List, predecessors list): These two columns store the identifiers of successors (predecessors respectively) of the concept.
- T_i,T_e (Size_Intension, Size_Extension): These two columns store the cardinality of a concept (respectively the number of attributes and the number of objects).

TABLE 3. SAMPLE CONCEPTS TABLE

C#	Niv#	N#	Int#	Ext#	L_s#	L_p#	T_i	T_e
....

E. Coursing the sample concepts table and calculating aggregation

In this step, we course the sample concepts table to extract approximate answers and to calculate the final result of approximate aggregation.

We use algorithm proposed in [16] to build a sample concepts table on the approximate query and then return the approximate answers. In order to improve the response time, we build the concepts table using only the query conditions.

This reduces the table size and minimizes the complexity of the construction of the sample concepts table.

We calculate the aggregation function (AVG, SUM and COUNT), using the algorithm 2 with the following descriptions:

- value (t): represents the aggregate value of the tuple t.
- degree (t): represents the membership degree of t.

To calculate the aggregation, we use these functions:

- For AVG() function :

$$AVG = \left(\frac{1}{n}\right) degree * \sum_{i=1}^n v(Li) \quad (1)$$

- For SUM() function:

$$SUM = degree * \sum_{i=1}^n v(Li) \quad (2)$$

- For COUNT() function:

$$COUNT = degree * \sum_{i=1}^n 1 \quad (3)$$

Where $degree = Min(U_{i1} \wedge V_{i1} \wedge \dots \wedge Z_{i1})$, and U, V, Z are the membership degrees on the query Q and n is the sample size, $v(Li)$ is the value of the tuple index i (L_i is a random index).

We calculate the error rate (Interval) associated with the aggregate function. We use the method of conservative confidence intervals [11]:

$$Error Rate = (b - a) \left(\frac{1}{2n} \ln \left(\frac{2}{1-p} \right) \right)^{1/2} \quad (4)$$

Where $[a, b]$ is a predetermined interval, such that $a \leq v(i) \leq b$ for all $1 \leq i \leq m$, n = sample size, m = size of KB, p is the setting of confidence (example $p = 0.95$).

Algorithm 2 : Calculate_function	
Inputs:	concepts table: TCX Maximum value of attribute :max Minimum value of attribute :min Sample size :n Aggregate function: f
Outputs :	result : res, Error rate :rate
Begin	
	D=1 som=0 card=0 For each element E of the concept table TCX if extension $\neq \emptyset$ then for each objet t of the extension som=som+value(t) card=card+1 if degree(t)< D then D=degree(t) End if End for End if End for If f= avg then res=(som/Card)*D else if f=sum then res=som*D else if f=count then res=card*D end if rate= $\frac{1,22 * (max - min)}{\sqrt{n}}$
End	

V. ILLUSTRATIVE EXAMPLE

Let a simple relational table “employee” (id, name, age, salary), which contains the following rows (see Table 4).

TABLE 4: EXAMPLE OF THE RELATIONNAL TABLE EMPLOYEE

ID	Name	Age	Salary
1	MOHAMED	23	400
2	ALI	30	550
3	WALID	45	700
.....
10000	WAJDI	40	800

The relaxing attributes *Age-Young*, *Age-Adult*, *age-Low*, *Salary-Middle*, *Salary-High*, and KB which contains rows as shown in Table 5:

TABLE 5 : CLUSTERING DATA OF THE RELATION EMPLOYEE

ID tuple	Age-Young	Age-Adult	Salary-Low	Salary-Middle	Salary-High
1	0.7	0.3	0.6	0.4	0
2	0.5	0.5	0	1	0
3	0	1	0.1	0.6	0.3
...
10000	0.1	0.9	0	0.3	0.7

Then, we eliminate data with low membership degree by setting a user defined threshold, KB becomes as shown in Table 6:

TABLE 6 : CLUSTERING DATA OF THE RELATION EMPLOYÉE WITH A THRESHOLD

ID tuple	Age-Young	Age-Adult	Salary-Low	Salary-Middle	Salary-High
1	0.7	-	0.6	0.4	-
2	0.5	0.5	-	1	-
3	-	1	-	0.6	-
...
10000	-	0.9	-	-	0.7

Consider the following query for finding the average salary for young employees and low salary with a confidence level = 95%.

"Average Salary of Young employees and Low Salary with a degree of confidence= 95% "

"Select Avg(Salary) from employees where age IS Young and Salary IS Low"

The approximate query becomes:

"Select AVG (Salary), 0.95 as confidence, ConsAvgInterval(0.95) from employee where age IS Young and Salary IS Low"

We construct the sample (Table 7) according to the KB at the time of query execution.

TABLE 7. SAMPLE OF DATA

ID row	Age-Young	Salary-Low	Salary
1	-	1	400
20	0.8	-	900
520	-	0.9	430
32	-	0.8	460
10	0.6	-	780
.....
130	-	0.5	550

Then we generate a concepts table associated with the query as shown in Table 8.

With each given extension contains two attributes: The first is the degree and the second is the aggregated value.

Example: 20 (1, 380) the row 20, a degree is 1and its value is 380.

We repeat these steps until all the KB is treated either we get an error rate is very low to say the exact result is very close to either the user is satisfied with the outcome and conclusion the query execution.

TABLE 8 : SAMPLE CONCEPTS TABLE

C#	Niv #	N#	int#	Ext#	L #	L_p#
1	1	1	Young_A low_S	Ø	(1,2,1) (1,2,2)	0
1	2	1	low_S	1(1 ;400) 32(0,8 ;460) 520(0,9 ;430) 130(0,5 ;550)	(1,3,1)	(1,1,1)
1	2	2	Young_A	10(0,6 ;780) 20(0,8 ;900)	(1,3,1)	(1,1,1)
1	3	1	Ø	1(1 ;400), 32(0,8 ;460), 520(0,9 ;430), ,130(0,5 ;550) 10(0,6 ;780), 20(0,8 ;900)	0	(1,2,1) (1,2,2)

In Table 9, we present an example of results returned after the calculation AVG and error rate functions.

TABLE 9: RESULTS OF APPROXIMATE ANSWERS

Avg	Confidence	Error rate
400	95 %	0.06503
402	95%	0.06500
405	95%	0.06470
....
410	95%	0.0090

VI. COMPARATIVE STUDY

In this section, we present the essential idea of the main approaches to flexible querying the closest to ours. We specify each time different art studies conducted on these approaches. They differ mainly by the way used to find the values closest to those requested by the user and the formalism used to model uncertainty and imperfection of the real world.

The contributions of the approach Ounelli *et al.* [8] are important, including the TAH and MTAH concepts for modeling generalization and specialization hierarchies of concepts. In this approach, no modification of SQL is required, what constitutes an asset for the implementation of this approach. The user does not apply during the relaxation to make choices that can be hazardous.

In this approach, the relaxing attributes are set by the administrator of the DB. This is especially important that the proposed approach is aimed at end users with no specific and

detailed knowledge on the organization and the data they consult. It is easier for an expert to specify a *price* attribute of the DB table is relaxing and can be used with the terms “*low*”, “*comfortable*” or “*high*”.

However, this approach has limitations in the structures it uses. We mainly include: i) incremental maintenance of the KB relaxing attributes, ii) clustering of relaxing attributes without fixing a priori the number of clusters, iii) the problem of storage ,clustering and indexing MTAH, and iv) not taking into account aggregate queries.

In the approach of Sassi *et al.* [10], generated clusters for each relaxing attribute are not stored in the DBMS catalog. Thus, the maintainability of this meta-base is no longer a problem. Indeed, in order to draw the concept lattice, core of FCA, they must simply load an XML file that can retrieve all the information necessary to trace these lattices.

However, this approach has limitations in the structures they use. We mainly include i) the number of concepts generated, ii) the response time used to generate approximate answers, and iii) not taking into account aggregate queries.

The approach of Chettaoui *et al.* [9] allows the treatment of empty response to a flexible query. Thus, it detects the causes of failure and allows the generation of sub-queries and approximate answers.

Another advantage of this approach is that not changing the structure of SQL and thus benefit from the features of the DBMS.

However, this approach does not allow the use of linguistic modifiers in the query. This test is interesting since users typically use such linguistic terms and it does not take into account the aggregate queries.

The approach of Hass *et al.* [11] allows classical querying (Boolean) on broad comic returning relevant answers in the shortest time for aggregate queries. It aims to gradually give approximate results when the query execution until all data has been processed. Thus, the user observes the degree of progress of the response and controls the query execution. We are not obliged to wait several minutes for the query result.

The approach proposed in this paper combines the advantages of those mentioned above while overcoming the limitations they present.

Indeed, we can perform an aggregate query on large flexible DB while returning relevant answers in the short time and the error rate.

Table 9 presents a comparative study between the approaches mentioned above and ours.

TABLE9. COMPARISON BETWEEN DIFFERENT APPROACHES

	Aggregation	Sampling	Flexibility	Accuracy
Query Relaxation	-	-	X	-
Fuzzification of concepts lattice	-	-	X	-
Online AQP	X	X	-	X
Flexible interrogation by AQP	X	X	X	X

VII. EVALUATION

Figure 3 shows the main interface of the FLEXTRA system.

The responses appear in the table when executing the query. In this case, the user does not have to wait until runtime to have the final result. Indeed, after a while, the initial response and the error rate is displayed, until the user stops the calculation or that KB has been completely treated.

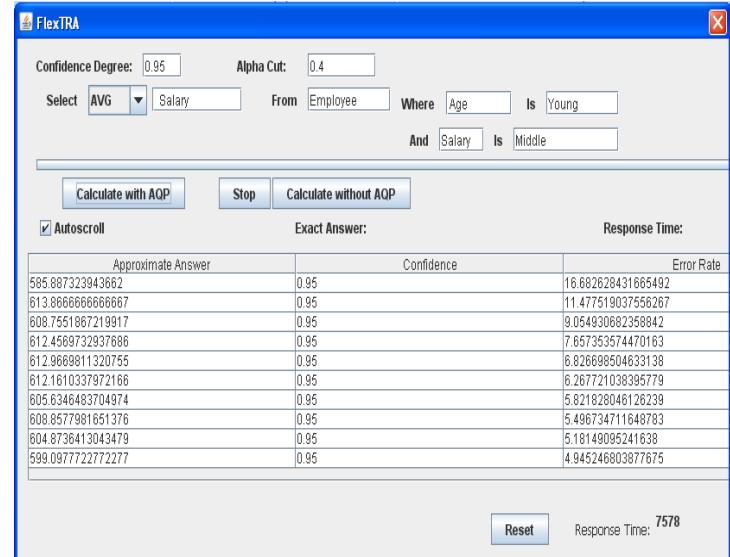


Figure 3. Display approximate answers

A. Testing the response time

We started with the table *employee* (*id*, name, salary, age) and we increased the number of records from 789 to 9498 records and calculate for each case the response time of FLEXTRA system and compare it with the case of AQ (approximate query) and classic querying(without AQ), as shown in Figure 4.

For the employee table, it contains two relaxing attributes “age” and “salary”.

The query is: “the average low salary of young employees”.

“*SELECT AVG (salary) FROM employee WHERE salary IS low AND age IS young*”.

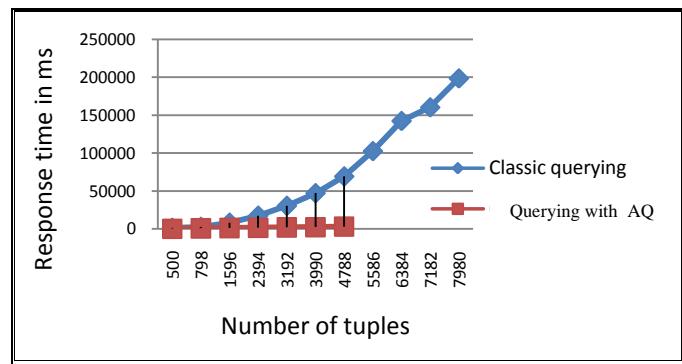


Figure 4. Comparison between the two approaches

From Figure 4, we find that the response time is lower using the AQ than classic querying.

For the classical query, the curve is exponential, while for interrogation with AQ, the curve is linear.

If the size of the database exceeds 7000 records, the response time for classic querying, is about 2 minutes, so it is with the AQ, the order of 5 seconds.

B. Testing the accuracy response

We will run the application in the table *employee* in both cases: with and without AQ, then check the quality of answers returned with AQ.

For the table *employee*, the exact answer is 34.9 years to have when we execute the following query:

“*SELECT avg(age) FROM employee WHERE age IS adulte AND salary IS middle*”.

Figure 5 shows the development of approximate answers to reach the exact answer:

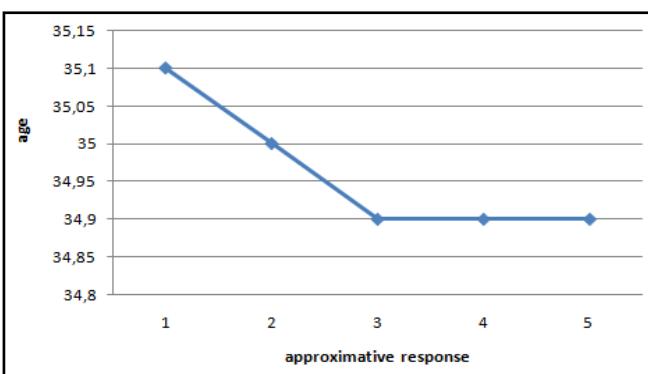


Figure 5. Evolution of approximate answers

We note that approximate answers are near to the exact answer, the first approximate answer is 35.1 years + / - 1.2 with 95% confidence level, whereas the correct answer (classic querying) is 34.9 years.

VIII. CONCLUSION

We proposed in this work a flexible querying approach of large DB while using AQ.

It is about a field in strong expansion. On the one hand, the DB are increasingly bulky. In addition, the construction of querying systems able to satisfy flexible queries is more complex and expensive.

We integrated the AQ techniques in a system using FCA in order to overcome limits of the existing approaches when we use aggregate queries (“Sum”, “Avg”, “Count”, “Var”, etc.) such as the response time and confidence rate of result answers.

The type of query is “give the average of the low salary”.

This system makes it possible to turn over quickly approximate answers while holding trying to improve the exactitude of the provided answers.

Our approach comprises two steps:

- Pre processing step in which the KB is generated starting from the DB so that it contains membership degrees of each tuple to the reliable attributes.
- Post processing step during which the flexible query is rewritten so that it becomes an AQ. The sampling of the KB consists in extracting some data (tuples). The construction of a sample concepts table is made to release the approximate answer and the Error Rate.

For the exactitude of the answer, we used AQ which support the response time to the detriment of the result exactitude.

In order to improve the response time, we propose in this article to adapt online aggregation technique proposed in [11], whose objective is to gradually give approximate answers while executing the query.

It consists to apply a sampling to the initial data of the DB in order to minimize the disc access and consequently to improve the response time.

Our approach contributes several shares in particular:

- The calculation of aggregation for flexible queries.
- The improvement of response time by guaranteeing the exactitude of the answer.
- The processing of the case of empty answers for a flexible query.
- No modification of the structure of the DBMS and SQL language.
- The use query execution control.

To implement this approach, layers will be added to a conventional DBMS such as:

- Rewritable layer: it takes care of rewriting the query aggregation for which an application becomes rough.
- Aggregation layer: it is responsible for calculating the responses gradually during the query execution.

As futures works, we propose:

- The integration of complex and nested queries involving join operation.
- The calculation of the aggregation functions on several attributes.
- The inclusion of some widely used language modifiers like “very” and “approximately” in the query qualification.
- The use of other sampling procedures in order to improve the confidence rate.

REFERENCES

- [1] Lacroix, M. and Lavency, P., “Preferences : Putting More Knowledge Into Queries”, 13th VLBD Conference, pp. 217-225, 1987.
- [2] Chan, C.L., “Decision Support in an Imperfect World”, Research Report, pp. 100-102, 1982.
- [3] Rabitti, F., “Retrieval of Multimedia Documents by Imprecise Query Specification”, Lecture Notes in Computer Science, 416, pp. 202-218, 1990.
- [4] Ichikawa, T. and Hirakawa, M., “ARES: A Relational Database with the Capability of Performing Flexible Interpretation of Queries”, IEEE Transactions on Software Engineering, pp. 624-634, 1986.

- [5] Motro, A., "VAGUE : A User Interface to Relational Database That Permits Vague Queries", ACM Transaction off Information Systems, 6(3), pp. 187-214, 1988.
- [6] Bosc, P., Liétard, L., and Pivert, O., "Bases de Données et Flexibilité : Les Requêtes Graduelles", Techniques et Sciences Informatiques, 17(3), pp. 355-378, 1998.
- [7] Tahani, V., "A conceptual Framework for fuzzy Query Processing: A step Toward Very Intelligent Database Systems", Information Processing and Management, 13, pp. 289-303, 1977.
- [8] Ounalli, H. and Belhadjahmed, R., "Interrogation flexible et coopérative d'une BD par abstraction conceptuelle hiérarchique", INFORSID 2004, pp. 41-56, 2004.
- [9] Chaettaoui, H., "Les treillis de concepts dans l'interrogation flexible et coopérative des bases de données", Master, Faculty of Sciences of Tunis, 2008.
- [10] Sassi, M., "Contribution à l'interrogation flexible des bases de données", Phd Thesis, National Engineering School of Tunis, 2007.
- [11] Haas, P.J., Hellerstein, J.M. and Wang, H.J., "Online aggregation". ACM-SIGMOD International Conference on Management of Data, pp. 171 - 182, 1997.
- [12] Chu, W.W., Yang, H., Chiang, K., Minock, M., Chow, G., and Larson, C., "CoBase : A Scalable and Extensible Cooperative Information System", Journal of Intelligent Systems, Kluwer Academic Publishers, vol. 6, Issue 2-3, pp. 223 – 259, 1996.
- [13] Olken, F., "Random sampling from databases", PhD Thesis, University of California, Berkeley.1993.
- [14] Chakrabati, S., Garofalakis, M., Rastogi, R., and Shim, K., "Approximate query processing using wavelets", Springer publisher, pp. 199 – 223, 2001.
- [15] Zadeh, L.A., "Fuzzy Sets", Information and Control, 8, pp. 338-353, 1965.
- [16] Gammoudi, M.M., "Décomposition conceptuelle des relations binaires et ses applications", Habilitation in computer science, Faculty of Sciences of Tunis, 2005.

IMA: Identification of Multi-author Student Assignment Submissions Using a Data Mining Approach

Kathryn Burn-Thornton

Data Mining Group

Brunel University

Uxbridge, UK

e-mail: Kathryn.thornton@brunel.ac.uk

Tim Burman

Dept. Computing and Engineering Science

Durham University

DURHAM, UK

e-mail: tim.burman@dur.ac.uk

Abstract—In this paper, we describe a novel application of data mining techniques which can be used to identify multi-authorship contained within student submissions. We show that by regarding the pages of the submission as a set of Cascading Style Sheets, CSS type files, which we call author signature styles (ASSs), and accompanying information, it is possible to identify the number of author signature styles contained within the page, or document, irrespective of the number of pages concerned. We also describe how, as a by-product of this work, a set of author signature styles (ASSs) can be created during investigation of each submission and hence be used as a library, containing increasing membership, for comparison with future submissions by the same student. The implications of the use of ASSs for identification of future suspect submissions, and for comparison with future submissions by the same student, are discussed.

Keywords-plagiarism; data mining.

I. INTRODUCTION

Government cuts in Higher Education funding have provided the driver for larger university class sizes, both face-to-face and online [19]. For class sizes greater than 50 this can mean that those marking essay style submissions may be unaware of the written style of the students and, in many cases, unable to put a name to a face [20-21]. For online students, the lecturer, or marker, may not ever meet the student [9, 13].

This lack of knowledge on the student puts the marker at a great disadvantage and provides a window of opportunity for those who are aware of the situation and who are keen to reuse material which may have been created by others i.e., those who are willing to plagiarize existing material. Such activity is readily facilitated by the virtual society which now makes it possible for students to access material from all over the world and with which the marker may not reasonably be expected to be aware [21].

Approaches to ameliorate this problem include continual assignment subject changes but it is not possible to ensure that they do not overlap with others set somewhere else in the world [20]. However, identification of whether the student's submission contains a duplication of information which may be found elsewhere on the superhighway is an approach to solving the plagiarism problem, which may be ideally suited to a software tool [2].

Although identical duplicate documents to those of student submission, or paragraphs, which may be readily found by making use of a simple search engine [24], submissions which contain modification of documents from various sources are harder to detect by this approach and a more sophisticated approach must be used to identify these. This has resulted in a 100 fold growth, over the last ten years, in published papers which outline approaches, and software tools, which may be used to provide aid in the detection of student plagiarism by universities [23].

However, the results from the use of plagiarism tools are often hard to follow using formal university procedure because of their determination of degree of commonality between the student submissions and other documents which are available [24-27]. In addition, the tools do not necessarily provide the user/investigator with an indication of whether, or not, the submission is individual original work. An approach which has not been used to identify 'suspect' student submissions, which may emanate from more than one author, is document signature style. This approach has an added advantage in that it is easier to follow up the results obtained using formal university procedures if required.

This paper describes a novel data mining approach, which enables documents to be identified which contain more than one document signature style. The first section describes current approaches which are used to identify 'suspect' student submissions.. This is followed by a discussion of two possible solutions which would enable document signature styles to be determined and a description of techniques which may be employed in order to achieve each of the potential solutions. Algorithms which may be gainfully employed in achieving the chosen solution are then outlined. The remaining sections discuss the investigations which were carried out in order to determine the effectiveness of the approach and the results of the investigations for the CSS type solutions – the ASS based solution. Conclusions regarding the results of the investigations are then drawn with future profitable avenues for investigation being discussed.

II. EXISITING APPROACHES AND TOOLS

The vast majority of tools, in common use in a university environment, which enable the investigation of submission of non-original work such as TurnitinUK and Viper [23, 27] appear to make the simplest assumption that

the submission of non original work by a student falls into the category of potential plagiarism. With this prior assumption that determination of plagiarism may be achieved by comparing the student's submission with all other submissions, and documents, which are available throughout the world.

The process is readily suited to current pattern matching algorithms, and methods, especially if paragraphs similarity between documents is to be considered. It is a type of pattern matching engine which underpins plagiarism tools which are commonly used in a university environment to identify potential plagiarized submission [23, 27].

Despite their speed of document comparison most tools of this type present the user(investigator) with a problem. Namely, that unless the submission is a 'simple' combination of existing work many of the current plagiarism tools do not provide sufficiently large a percentage match between the student's submission and documents which may be available on the web in order to pursue further the investigation of the lack of originality in the submission using formal university approaches [29]. However, another approach to detection of non-original work in the student submission could prove profitable when the pattern matching approach fails, that of the author signature style [6, 12, 14, 23] (ASS) since all student submissions should emanate from one student so should contain only one ASS or a variant on the same ASS.

III. DOCUMENT SIGNATURE STYLE

Document signature style makes the assumption that each individual has a unique writing style which is characterized by their individual use, and combination, of nouns, verbs and other features which include referencing [7, 12, 14, 23]. If the document signature style were to vary throughout a document's paragraphs, pages and chapters this could provide an indication that the submitted document originated from more than one author and was not the submission from one individual.

Such variation in style could be used as a basis for a formal university approach as the student submissions profess that the word is their individual work, in other words from only once source. This approach could, if sufficiently accurate, prove to enable the task to be achieved faster, and hence enable more student submissions to be checked, because all the information in cyberspace is not be trawled for each submission. The following section outlines how such a solution can be achieved.

A. Extraction of Signature Style

In order to determine the unique author signature(s) present in the electronic submissions it necessary to determine which key elements of written documents will be used to determine the unique documents signature created by each author. Initial analysis of over 300 submissions from this university [29] suggested that the key elements of the signature required in order to determine whether, or not, a document emanates from one author, may be reduced to number of words in a sentence, number of lines in a paragraph, paragraph formatting, degree and use of grammar, type of language used, word spelling and

referencing style. These key signature features are concomitant with those proposed at ICADPR for instance those in [7] and [10].

The first two elements of the signature are self explanatory but the others may require some clarification. Degree and use of grammar is taken to include the manner in which infinitives are used; use of, and types, of punctuation; use of plurality. Type of language is taken to mean language style which includes different types of English for instance UK and US. However, word spelling includes not only language spelling differences such as those found between UK & US, for example as in counsellor and counselor, but also frequency of typographical errors and spelling mistakes. The referencing style required by different bodies, and institutions, vary and can provide an indication of material which originates from more than one source.

A solution to this problem will be an approach which will enable extraction of the key signature elements, and their values, from paragraphs, and pages, and compare them with others in the same document and with those extracted from other documents. It would also be useful if the approach taken could show how the document would have appeared if written by a sole author if additional proof of multi-authorship was required for use in a formal university process.. The following section describes two possible solutions.

IV. POSSIBLE SOLUTIONS

Both of the solutions suggested in this section make use of approaches which we used in our web site maintainability tool [1]. The approaches make use of Cascading Style Sheets (CSS) or a combination of the eXtensible Markup Language (XML) in combination with the eXtensible Style Language (XSL) [17]. The approaches which we suggest make use of information extraction and representation. Some commonality can be observed between the first steps of the approaches, which are described in the next section, and that of Ghani [8] and Simpson [15].

A. CSS

If a CSS -based approach were used, a named author signature style (ASS) could be defined which would describe the values assigned to the key signature features. Once the ASS files were created, the signature of style of the author could not only be compared with others within the same document but it could also be applied to any document section and the output compared with that contained within the current, or other, submitted document. By using this approach the speed of investigation of submitted documents could be minimized by the reduction in the size of file which is required in order achieve comparison [16].

In practice, each section of the document being investigated could be converted directly to a section of ASS containing the feature values. Such an approach would require the use of a measure of uncertainty when mapping the samples of document and related ASS code to named signature styles. Figure 1 provides an example of how a page of text may be converted using such an approach.

Data mining would appear to provide a solution to this problem using clustering techniques.

The only drawback to this approach is that a library of assignable values for each key signature feature will need to be defined initially. However, this library may be updated as each submission is investigated.

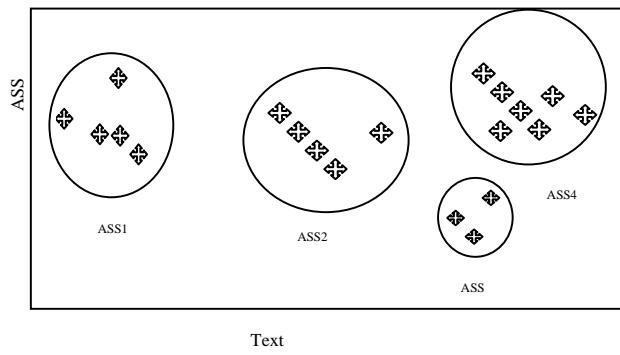


Figure 1. Clustering.

B. XML

For an XML approach all content information would be contained in an XSL file with its companion XML file containing the ASS feature information which would be recursively applied to the XSL document.

Using the example from Figure 1 this approach would result in the production of a XML file containing a section of text that would be marked up as a reference name, and the XSL file would contain a template which could be applied reference names in that document. Such an approach would readily facilitate comparison of documents because it would be relatively easy to target comparison of documents by investigation of specific signatures, ASSs.

Rigid definitions do not exist for XML tags which means that any appropriately defined names will have to be used in the XML file as well as a library of attributable values of the signature features, as in the CSS approach. However, a major drawback of this approach would be the need of consistency for XML tags and the possibility of ongoing modification to a centrally accessed XML tag dictionary.

Both requirements for the XML/XSL approach suggest that the CSS based solution may be more accurate to carry out comparison of signature styles in documents because even a slight variation in XML tags could result in a large discrepancy in ASSs and hence identify a document as containing information from more than one author when it does not.

The following section provides an introduction to data mining, the basis of the CSS , or ASS, approach.

V. DATA MINING

Data mining finds novel, potentially useful and ultimately understandable patterns from mountains of data [3] and has been used to mine data from diverse domains including the medical domain [5], pharmaceutical [4] and, as such, appears to be the ideal solution for finding the

patterns of information contained within the files extracted from (and contained within) the student submissions. This is an approach which we used in our web maintainability tool [1].

Data mining determines the patterns by clustering the data according to variable values contained in the data [11]. Figure 1 shows how clustering could be carried out using pre-determined CSS, or ASS, files and unclassified student submissions. In this example – each sample in the classifier is marked with a cross indicating the document page giving rise to the sample, and the CSS section (ASS section) that was generated from the document. Samples that have similar values, appear to have been produced by the same author, are given the same classification.

In clustering, each CSS section would be classified in turn. If it is sufficiently similar to other previously classified sections, ASS, it is added to the same classification (class) as these other sections. If it is not sufficiently similar to another section, a new classification, a new ASS, is created.

There are many different classes of Data mining algorithm which can perform clustering with each class possessing different properties. It is these different properties which make each class suitable for analyzing different types of data [11]. The class of algorithms which appear to be particularly appropriate for mining the type of data of which CSS files are composed belong to the statistical & machine learning classes of algorithms. More information regarding this may be found from the results of the STALOG project [23]. These classes of algorithms are described, briefly, in the following section.

A. Suitable data mining Algorithms

The suitability of algorithms chosen from the statistical & machine learning classes, namely: - k nearest neighbours, linear (k-NN), quadratic & logistic discriminants, k means, rule based, decision trees and Bayesian classifiers are described and their appropriateness for the task in hand. These are the same algorithms which were discussed for the task of web site maintainability [1].

The most appropriate algorithm for the conversion from student document to CSS, ASS, from those listed above, is the k-NN algorithm. The other algorithms are not appropriate because they either require too many samples with which to build an effective model from which to work effectively in this application (decision trees, Bayesian classifiers), require numerical data (Fisher's linear discriminants), or require prior knowledge of the classes (K Means). However, k-NN can work effectively with a small number of samples, can work with categorical data given an appropriate function to compare two samples, and does not require any prior knowledge of the number of classes, or authors.

The following sections describe the implementation of the CSS solution which has been described in this section.

VI. CSS SOLUTION

In order to implement the k-NN algorithm some means of finding a numeric difference between two samples of student document and ASS is required. This can be achieved by determining the percentage of elements of the code in one

sample which is not present in the other sample. In order to determine the difference between the two samples of ASS signature features, and their values, present in each sample signature needs to be investigated. A visual representation of this process may be seen in Figure 2.

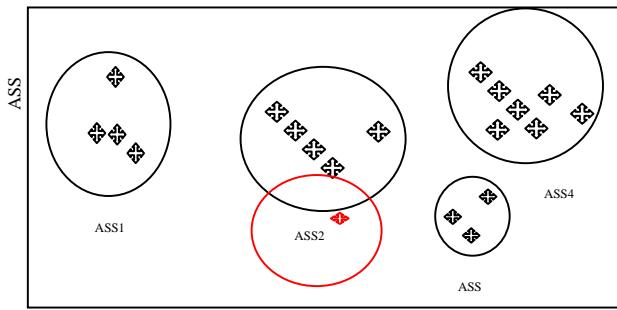


Figure 2. Clustering using K-NN.

In order to achieve this each section of student document, submission need to be represented by equivalent signature features and their values. In the same manner as presentation tags in HTML code these can be represented as signature tags. It is these adjacent signature tags which form clusters of tags and can be represented by a single ASS.

The first stage of the implementation of the k-NN algorithm is to create the signature tags from the original document and then each cluster of signature tags is converted to a ASS sample using a set of rules that are defined in a data file. This can be changed by the user as the ASS evolves, but a standard set of rules.

Each line is in the format:

Tag-name	ASS-equivalent	value
----------	----------------	-------

After each cluster is converted to an ASS the algorithm iterates through each sample and compares it to any that have already been classified. At the start of the loop, none will have been classified. Otherwise, a list of the other classified samples is created and ordered by difference to the new sample. If no sample is within a threshold distance, it is assumed that the new sample is not sufficiently similar to any previous classification, and so the user is prompted for a new classification for this sample. Otherwise, the closest k samples are taken from this list and the new sample is assigned the same classification as the majority of these k samples. An appropriate value of k can be found through trial and error during the implementation.

For the final conversion of the classifications to a style sheet, an arbitrary sample from each classification is used to supply the definition of the style, and the name assigned to the classification is used as the name of the style. As each sample in the class should be very similar, it should not matter which sample is used for the style definition.

A slight modification was made to the k-NN class so that it could be used to create an example document from an existing signature style. This modification was that a new author signature is not created if no close match among the previously classified samples is found. The contents of the style sheet are read in and set as the classified samples to provide the classification. The same approach is used for finding groups of pages with the same style. The major differences in this case is that the methods used to represent each page, and the differences between them – as well as the automatic naming procedure of a process which is to all intents and purposes completely unsupervised.

Each page, or paragraph, is represented by a set of feature information, including a list of the number of times each one is used, and the distribution of the feature tags throughout the page or paragraph. The combination of this set of information gives a good overall impression of the written signature style of the author.

The difference between two sets of information is found by the number of features, and values, that are not present in one set of information and is present in another, or those where the font or style is used more than twice as many times in one than in the other. The table distributions are compared using the chi-squared test. Each distribution is composed of 100 values, indicating the number of signature tags in that 100th of the section. The chi-squared value is calculated as the sum of the squares of the differences of each of these values, as given by the formula:

$$\chi^2 = \sum_{i=1}^{100} \frac{(x_i - y_i)^2}{y_i} \quad (1)$$

where x is distribution of table tags in Section 1
y is distribution of table tags in Section 2.

The set of this information provides an overall value for the difference between the two pages. This can then be directly compared to the value for any other two pages. Again, if the page being classified is not sufficiently similar to any previously classified section, a new classification, or ASS, is created for it.

The following section describes investigations which were carried out in order to determine the effectiveness of the CSS methods to facilitate comparison of author signature styles (ASS) in the paragraphs comprising the students

VII. INVESTIGATIONS

In order to determine the effectiveness of the solution, a set of metrics were defined which enabled the effectiveness of the solution to be determined on a wide range of submitted documents.. This section describes the metrics used and the wide range of documents used.

A. Measures of Effectiveness: Metrics Used

The effectiveness of the solution was determined by the ease, and effectiveness, of extraction of file information from the source page into a separate author signature style sheet

and the degree to which the content of the original pages remained unaltered once it has been re-produced by use of the style sheet.

Two metrics were also used to determine the effectiveness of the solution. Firstly, relative time for comparison of author signature styles in paragraphs contained within student submission by tool with that taken by a human carrying out the same task. Secondly, the number of author signature styles produced and number of differences between the signature style features in the original page, or paragraph, in the submission and that created using the ASS Documents Investigated

These submissions were chosen as examples of their wide range of document pages to which the algorithms, which represent the algorithm of the tool, can be applied because they represent a cross section of the variation in author styles contained with documents submitted at this university.

Sample	Student Origin	First Language	Number	Authorship
1	UK	English	20	Known Single
2	UK	English	20	Assumed Single
3	EU & UK (50:50)	50 English: 50 2 nd English	20	Assumed Single
4	Not EU	Not English	20	Assumed Single
5	UK	English	20	Known Multi

Figure 3 provides examples of the wide range of student submissions which were investigated.

Sample 1 containing documents known to have been written by one author. Sample 2 contains UK students whose first language is English whilst Sample 3 contains an equal mix of EU and UK students. Sample 4 contains non EU students who are required to take TOEFL and who have all passed the level required to be admitted to the university. Sample 5 contains documents which are known to contain multiple authorship.

This range of documents should enable the performances of the algorithm, and hence tool, on different written styles of pages to be determined.

The following section describes the results from applying the metrics to the wide range of test documents.

Figure 3. Examples of submission types.

VIII. RESULTS

Simple plots are used to visualize the results. Figures 4 to 6 show the results of investigation of the three metrics.

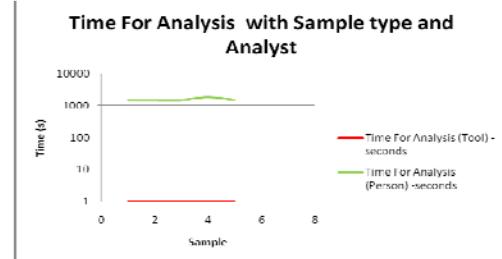


Figure 4. Relative time for paragraph authorship comparisons.

A. Relative time for paragraph authorship comparisons

The results of these investigations are shown in Figure 4. The figure shows that the tool was able to perform comparison up to 1000 times faster than the person carrying out the same task. The figure also shows that the time taken for the non-tool based comparison of the signature style in each paragraph varied from person to person and also from sample type to sample type. There was no difference in the comparison time for the tool because of the short time in which this was achieved, all within 1 second.

Half of the of people carrying out the task were unable to complete the comparison for any of sample size 4 because of the fluency in the written style of the student submissions.

B. A count of the author signature styles produced

The number of signature styles produced is dependent of the written content of each page. Figure 5 shows that, on average, two styles are produced from a page known to have been written by one author. The figure also shows that, on average, three styles are produced from a page of unknown authorship with the distribution of the number of styles produced being skewed towards the lower end. The tool accurately determined the number of the authors from the documents known to be multi-author. However, the figure shows that human determination was less accurate – especially for samples 3 and 4, those for which English was not a first language.

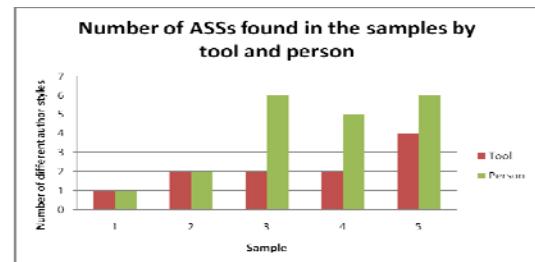


Figure 5. A count of the author signature styles.

C. Information Differences

These results shown in Figure 6 are consistent with that results of the ASS investigations in that information differences observed between the original, and key features of the, document are strongly correlated with the error in

determining authorship number. Thus suggesting that if the ASSs contained in the document can be determined then it is possible to reform key features of the original document for comparison with other student submissions and with future by the same student.

IX. CONCLUSIONS AND FUTURE WORK

We have described an approach for carrying out investigation of the plurality of the authorship of documents submitted by students, which is dependent upon Data mining-based clustering methods.

The results presented in section VIII show that this approach facilitates accurate investigation of the authorship of student document submission. Such results have the potential to be used in formal university procedures.

Is intended that further work will be carried out investigating the three key metrics in submission from other Faculties and universities. Work will also be carried out to modify the Data mining algorithm to maintain accuracy of Multi Author Determine across this new range of submissions.

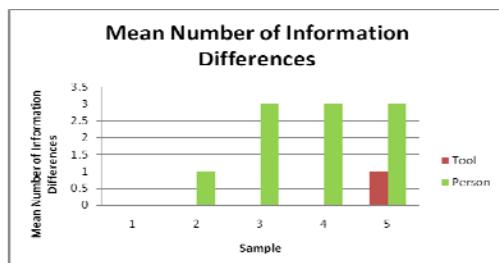


Figure 6. Information Differences between Original and Reformed Text produced.

ACKNOWLEDGMENT

Acknowledgment is made to Mark Carrington for his original project work in 2002 which led to development of this paper. The authors wish to thank the University of Durham for the provision of facilities to carry out this work.

REFERENCES

- [1] K. E. B. Thornton, M. Carrington, and T. Burman, "A Data mining based method for web site maintenance," Intelligent Data Analysis, vol. 10, 2006, pp. 555-581.
- [2] K.E. Burn-Thornton, D.M. Catrall, and A. Simpson, "Polymorphic Functions for Data mining in A.T.M. Networks," Proc. 4th I.F.I.P. Conf., July 1996, pp.11-18, Ilkely, UK.
<http://www.xent.com/summer96/0041.html>, accessed 6/12/10.
- [3] K.E Burn-Thornton., S.I. Thorpe, and J., A Attenborough,"Method for Determining Minimum Data Set Size Required for Accurate Domain Analysis." in Proc. PADD '00, International Data mining Conference, May 2000, pp. 161 –169, Manchester - ISBN 1 902426 08 8.
- [4] K.E. Burn-Thornton and J. Bradshaw," Mining the organic compound jungle – a functional programming approach," chapter 11, IEE Practical Applications of Computing, March 1999, pp. 227-240.
- [5] K.E. Burn-Thornton and L. Edenbrandt, "Myocardial Infarction - Pinpointing the Key Indicators in the 12 lead ECG Using Data mining," Computers and Biomedical Research, vol. 31, 1998, pp. 293-303..
- [6] J. Cai, R. Paige and R. Tarjan,"More Efficient Bottom-Up Multi-Pattern Matching in Trees," Theoretical Computer Science, vol. 106, pp.21-60, 1992.
- [7] C.E. Chaski,"Multilingual Forensic Author Identification through N-Gram Analysis," Paper presented at the annual meeting of the The Law and Society Association, TBA, Berlin, Germany 2010-06-04 from http://www.allacademic.com/meta/p177064_index.html, accessed 17/10/10.
- [8] R. Ghani, R. Jones, D. Mladenic, K. Nigam, and S. Slattery,"Data mining on symbolic knowledge extracted from the web," in Proc. of the Sixth International Conference on Knowledge Discovery and Data mining (KDD-2000), Workshop on Text Mining, pp. 21-29.
- [9] J. Hewitt and C. Brett," The relationship between class size and online activity patterns in asynchronous computer conferencing environments," Computers and Education, vol. 49, pp. 1258-1271, 2007.
- [10] B. Kövesi, J.M. Boucher, and S. Saoudi, "Stochastic K-means algorithm for vector quantization," Pattern Recognition Letters, vol. 22, pp. 603-610, 2001.
- [11] D. Michie, D.J. Spiegelhalter, and C.C. Taylor (ed),"Machine learning, neural and statistical classification," New York: Ellis Horwood, 1994.
- [12] A. Brink, L. Schomaker, and M. Bulacu,"Towards Explainable Writer Verification and Identification Using Vantage Writers," in. Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), pp. 824-828, Parana, Brazil September 23- 26, ISBN: 0-7695-2822-8.
- [13] N. Shadbolt, "Caught up in the web," Invited talk at the 13th International Conference on Knowledge Engineering and Knowledge Management, EKAW02., 2002, pp. 317-334.
- [14] I. Siddiqi and N.Vincent, "Writer Identification in Handwritten Documents," Proc. Ninth International Conference on Document Analysis and Recognition (ICDAR 2007) in Proc. Document Analysis and Recognition, International, 2007, vol. 1, pp. 108-112.
- [15] S.Simpson,
<http://www.comp.lancs.ac.uk/computing/users/ss/websitemgmt>, accessed 6/12/10.
- [16] I. Sommerville, "Software engineering," 5th ed., International computer science series, Wokingham, England : Addison-Wesley, 1996.
- [17] Wilde E, Wilde's WWW. Technical foundations of the World Wide Web. London: Springer, 1999.
- [18] ZigZag,www.zigzagdesign.co.uk/website_maintenance.htm, accessed 6/12/10.
- [19] <http://www.hefce.ac.uk>, accessed 6/12/10.
- [20] <http://www.alluniversities.com/index.php>, accessed 6/12/10.
- [21] <http://www.articlesnatch.com/Article/Uk-Academic-Writing-Service/1239456>, accessed 6/12/10.
- [22] <http://www.euroscience.org/author-identification,28115,en.html>, accessed 6/12/10..
- [23] <http://www.google.co.uk> accessed 6/12/10
- [24] www.scammyessay.com accessed 6/12/10.
- [25] cs.stanford.edu/~aiken/moss/ accessed 6/12/10.
- [26] turnitinUK, www.submit.ac.uk/, accessed 6/12/10.
- [27] <http://www.brun.ac.uk> , accessed 6/12/10

A Representation of Certain Answers for Views and Queries with Negation

Victor Felea

Department of Computer Science

"Al. I. Cuza" University

Iasi, Romania

Email: felea@infoiasi.ro

Abstract—The paper is about databases content processing, namely query processing. Certain answers are very important in the study of the data complexity of the problem of answering queries using materialized views and constitute a semantics of query answers in mediated integration systems. The computing of these answers depends on database and view models, and the case when the negation occurs was not studied. In this paper, we give a representation of certain answer sets where both the query and the views are expressed in conjunctive form with negation. Using this representation, a method to compute the certain answers for the open-world and close-world assumptions is given.

Keywords—views; queries; negation; certain answers.

I. INTRODUCTION

In many data-management applications such as data-integration from different sources, data warehousing, query optimization, the problem of view-based query processing is central. The study of this problem implies the consideration of several main notions as *answering* and *rewriting*. *Answering* means the computing the tuples satisfying the query in all databases consistent with the views. *Rewriting* is a reformulating of the query in terms of the views, and then evaluating the rewriting over the views extensions. In general terms, the problem of rewriting is as follows: given a query Q on a database schema S , expressed in a language \mathcal{L} , and a set \mathcal{V} of views on S , can we answer Q using only \mathcal{V} ?

A lot of results have been reported in the last years and many methods have been studied(see a survey in [1]). One of the approach of view-based query processing problem is the query-answering approach, where so-called certain tuples ([2]) are computed. Certain tuples are the tuples that satisfy the query in all databases consistent with the views, on the basis of the view definitions and the view extensions.

In [2],[3], some aspects and applications of the problem of answering queries using views, and algorithms are presented. Some authors study the problem of view-based query processing in a context where databases are semistructured, and both the queries and the views are expressed as regular path queries in [4],[5]. A tableau technique is used for computing query answers in [6]. In [7], the authors study the complexity of query answering considering key and inclusion dependencies. The problem of answering queries using views, when queries and views are in conjunctive form with arithmetic comparisons, are analyzed in [8].

The structure of the paper is following: in Section II, we

discuss the main papers concerning to the computing of certain answer sets, in Section III, we specify the basic definitions and notations used in the paper. In Section IV, we give a representation of certain answer sets in case of open-world assumption. A method to compute certain answer sets in cases of open-world and close-world assumptions is given in Sections V and VI, respectively. The problem of time complexity to compute certain answers is analyzed in Section VII. Finally, Section VIII concludes the paper.

II. STATE OF THE ART

In the relational model, a query Q_1 is said to be contained in the query Q_2 if Q_1 produces a subset of the answers of Q_2 , for any database. In the context of data integration, we say that Q_1 is contained in Q_2 relative to a set of views \mathcal{V} if, for any set I of instances of \mathcal{V} , the certain answers of Q_1 are a subset of the certain answers of Q_2 . In [2], the authors study the complexity of computing certain answers in case when views and queries are in conjunctive form, conjunctive form with inequality, non-recursive datalog, datalog or first order formula. In case when the query is expressed in datalog, and does not contain comparison predicate, and the views are in conjunctive form, the set of certain answers can be obtained by so called a query plan, which is a datalog program whose extensional relations are the source relations. More precisely, the maximally contained query plans defined in [9] compute the certain answers of the query [2]. In [10], using certain answers, the authors define *relative containment*, which formalizes the notion of query containment relative to the sources that occur to the data-integration system. In [11], the containment of two queries is studied. In [5], the authors study the problem of answering queries using materialized views in the presence of negative atoms in views. By our best knowledge, the problem to compute the certain answer set in case when the negation occurs in views or query was not considered yet in literature.

Let us give a motivated example.

Example 1 Let us consider the relational schema \mathcal{S} consisting of $\{COMP, CON, PROD, ITEM\}$, where $COMP$ represents the companies and has *comp-id* as an identification code of a company, and *comp-name* is the name of the company,

CON represents the contracts between companies, and has

as attributes *con-id* the identification code of the contract, *b-comp*, the beneficiary of the contract *con-id*, and *f-comp* is the supplier of the contract *con-id* (the values of *b-comp* and *f-comp* are company codes),

PROD represents the products, and has *prod-id* as the identification code of a product, and *prod-name* the product name,

ITEM represents items specified by the contracts, and has as attributes: *con-id1*, the code of a contract, *prod-id1*, the code of a product. We consider the following view: find contracts x_1 , companies x_2 , and products x_3 such that x_2 is the beneficiary company of x_1 and the product x_3 occurs as item of the contract x_1 , and there exists a contract y_2 such that the product x_3 does not occur in the contract y_2 .

Using the schema \mathcal{S} , we can express this view definition as:

$$V(x_1, x_2, x_3) : -CON(x_1, x_2, y_1), CON(y_2, y_3, x_2),$$

$$PROD(x_3, y_4), ITEM(x_1, x_3), \neg ITEM(y_2, x_3),$$

where the character ‘,’ between two literals means logical conjunction. Let us consider the query: find all companies z such that there exist two contracts t_1 and t_3 , where one of them contains z as the beneficiary and another as supplier and there exists a product t_5 such that one or another from contracts t_1 and t_3 does not contain the product t_5 . We express this query as follows:

$$Q : q(z) : -CON(t_1, z, t_2), CON(t_3, t_4, z),$$

$$PROD(t_5, t_6), (\neg ITEM(t_1, t_5) \vee \neg ITEM(t_3, t_5))$$

It is clear that the query Q is equivalent with a union of queries in conjunctive form. Let I be an extension of V , where $I = \{\bar{w}_1, \bar{w}_2\}$, and $\bar{w}_1 = (1', S2', P2')$, and $\bar{w}_2 = (2', S3', P3')$. We are interested to compute the certain answers corresponding to I , V and Q .

In this paper, we compute the certain answer sets in two cases: under the open-world assumption (*OWA*) and under the closed-world assumption (*CWA*) in case when views are in conjunctive form, and query is a union of conjunctive form, and both can contain negative literals.

III. BASIC DEFINITIONS AND NOTATIONS

Let Dom be a countable infinite domain for databases. A view definition has the following form:

$$\begin{aligned} V(\bar{x}) : & -R_1(\bar{u}_1), \dots, R_k(\bar{u}_k), \neg R_{k+1}(\bar{u}_{k+1}), \dots, \\ & \neg R_{k+p}(\bar{u}_{k+p}), \end{aligned} \quad (1)$$

where \bar{x} is a vector of variables. These variables are called free in the view V . The vectors \bar{u}_i consists of variables or constants, $1 \leq i \leq k + p$. All constants are considered in Dom . There are two restrictions about variables or constants that occur in the view. The first one is: each variable that occurs in \bar{x} , it must appear also in at least a vector \bar{u}_i , $1 \leq i \leq k$, that means it also appears in the positive part

of the view definition. This is called the safe property of the view. The second one is: each variable or constant that occurs in the negated part of the view definition, must occur in its positive part. This property is called as safeness property of negation. The symbols R_i are relational symbols, $1 \leq i \leq k + p$. All variables from \bar{u}_i , $1 \leq i \leq k + p$, that are different from variables from \bar{x} , are called existentially quantified variables. Let us denote by $f_V(\bar{x}, \bar{y})$ the right part of the view definition V . A query Q in conjunction form with negation has a similar form as view definition. Let us denote by $q(\bar{z})$ the head of the query, and $f_q(\bar{z}, \bar{t})$, the right hand part of the query, where \bar{t} denote all existentially quantified variables from the query. In an integration system the views are called sources. If sources have non-relational data models, we can use *wrappers* [12] to create relational view of data. In the following definition we present the notion of certain answer in two cases: (*OWA*) and (*CWA*).

Definition 1: Let Q be a query and $\mathcal{V} = \{V_1, \dots, V_m\}$ be a set of view definitions over the database schema $\mathcal{S} = \{R_1, \dots, R_s\}$ (all relational symbols from \mathcal{S} are used in at least V_i). Let \bar{w}_i be an extension of the view definition V_i , for each i , $1 \leq i \leq m$. Let $I = \{\bar{w}_1, \dots, \bar{w}_m\}$. The tuple t is a certain answer for I , \mathcal{V} , and Q under *OWA* if $t \in Q(D)$, for all databases D defined on Dom such that $I \subseteq \mathcal{V}(D)$. The tuple t is a certain answer for I , \mathcal{V} and Q under *CWA* if $t \in Q(D)$ for all databases D defined on Dom such that $I = \mathcal{V}(D)$.

In an intuitive sense, a tuple is a certain answer of the query Q , if it is an answer for any of the possible database instances, which are consistent with the given extensions of the views. Concerning the number of view definitions, and the number of extensions of the views from \mathcal{V} we distinguish three cases: (I) \mathcal{V} consists of a single view definition denoted V , and I consists of m extensions of V , denoted $\bar{w}_1, \dots, \bar{w}_m$.

(II) \mathcal{V} consists of multiple definitions of a view V , denoted $V(\bar{x}) : -f_V^i(\bar{x}, \bar{y})$, $1 \leq i \leq h$, and I consists of m extensions of the view definitions of V .

(III) \mathcal{V} consists of multiple view definitions of the views V_1, \dots, V_q and I consists of m extensions, an extension corresponds to a view definition of any view V_i , $1 \leq i \leq q$.

For the sake of the presentation, let us consider the case (I), the approaches of the cases differ only in notation.

In the case of the open-world assumption, we express the relation $I \subseteq \mathcal{V}(D)$ that it is equivalent to: $\bar{w}_i \in V(D)$, for each i , $1 \leq i \leq m$, where D is a database defined on Dom . Then there exists a mapping ν from the set of all variables from \bar{x} and \bar{y} into Dom such that the following relations yield:

$$\nu pos(f_V(\bar{x}, \bar{y})) \subseteq D, \nu neg(f_V(\bar{x}, \bar{y})) \cap D = \emptyset, \nu \bar{x} = \bar{w}_i, \quad (2)$$

where $\nu(c) = c$, for a constant c ,

$$pos(f_V(\bar{x}, \bar{y})) = \{R_1(\bar{u}_1), \dots, R_k(\bar{u}_k)\} \text{ and}$$

$$neg(f_V(\bar{x}, \bar{y})) = \{R_{k+1}(\bar{u}_{k+1}), \dots, R_{k+p}(\bar{u}_{k+p})\}$$

We denote by $Rel(f_V(\bar{x}, \bar{y}))$, the set of all relational symbols that occur in the conjunction $f_V(\bar{x}, \bar{y})$. For a mapping ν having the property from (2), we denote by $f_V(\bar{w}_i, \bar{y})$ the

result of replacing all free variables x' from \bar{x} with $\nu(x')$. For two different vectors \bar{w}_i and \bar{w}_j , the existentially quantified variables of type y are independent, hence we take the sets of variables for \bar{y} disjoint. So, we denote by $f_V(\bar{w}_i, \bar{y}_i)$ or f_i , the expression $f_V(\bar{x}, \bar{y})$, where \bar{y} is replaced by \bar{y}_i and the set of all variables from \bar{y}_i is disjoint from the set from \bar{y}_j , that means $\bar{y}_i \cap \bar{y}_j = \emptyset$ for all $i, j, 1 \leq i \neq j \leq m$. Let us denote by C the set of all elements from Dom , that appear in the vectors $\bar{w}_i, 1 \leq i \leq m$ and in $f_V(\bar{x}, \bar{y})$. Let Y be the set of all variables from $\bar{y}_1, \dots, \bar{y}_m$. Let π be a partition of the set $C \cup Y$, and $Class_\pi$ the set of the classes defined by the partition π . We denote by \equiv_π the congruence relation defined by the partition π , namely: we have $t \equiv_\pi t'$ if there exists a set M from $Class_\pi$ such that $t, t' \in M$. In the paper we only need partitions with the property: for two different constants c and c' , we have $c \not\equiv_\pi c'$. These partitions are called C -partitions. It is clear that for a C -partition a class contains at most a constant. For a partition π on $C \cup Y$, we consider a mapping from $C \cup Y$ into $Class_\pi$ denoted φ_π called the canonical onto mapping and defined by $\varphi_\pi(t) = [t]_\pi$, where $[t]_\pi$ means the class that contains t . The mapping φ_π is extended naturally to a vector $\bar{w}' = (t_1, \dots, t_r)$ on $C \cup Y$ by $\varphi_\pi(\bar{w}') = (\varphi_\pi(t_1), \dots, \varphi_\pi(t_r))$. For an atom $R(\bar{w}')$, we consider $\varphi_\pi(R(\bar{w}')) = R(\varphi_\pi(\bar{w}'))$. For a set of atoms S having the form $R(\bar{w}')$, we define $\varphi_\pi(S) = \{\varphi_\pi(R(\bar{w}')) | R(\bar{w}') \in S\}$. Associated to a C -partition π , we define two databases having elements from $Class_\pi$, and denoted T_π^{min}, T_π^{max} , in the following manner:

$$T_\pi^{min} = \bigcup_{i=1}^{i=m} \varphi_\pi pos(f_V(\bar{w}_i, \bar{y}_i)) \quad (3)$$

$$T_\pi^{max} = \{\varphi_\pi R(\bar{w}) | R \in Rel(f_V(\bar{x}, \bar{y})), \bar{w} \text{ on } C \cup Y\} - \bigcup_{i=1}^{i=m} \varphi_\pi neg(f_V(\bar{w}_i, \bar{y}_i)) \quad (4)$$

We denote by \mathcal{M}_π the set of all databases between T_π^{min} and T_π^{max} , i.e.:

$$\mathcal{M}_\pi = \{T | T_\pi^{min} \subseteq T \subseteq T_\pi^{max}\}$$

For a conjunction of literals f_i , we need to consider a formula denoted $\phi(f_i)$, whose basic elements have the form $(t_i \neq t_j)$, where t_i and t_j are elements from $C \cup Y$. Let f_i having the form: $f_i = R_1(\bar{z}_1), \dots, R_k(\bar{z}_k), \neg R_{k+1}(\bar{z}_{k+1}), \dots, \neg R_{k+p}(\bar{z}_{k+p})$. Let $R_{k+j}(\bar{z}_{k+j})$ be an atom that occurs in the negated part of f_i . Let us consider the case when R_{k+j} occurs in the positive part of f_i , with the indexes $\alpha_1, \dots, \alpha_q$, that means we have: $R_{k+j} = R_{\alpha_1} = \dots = R_{\alpha_q}$, and $R_{k+j} \neq R_\beta$ for each $\beta \in \{1, 2, \dots, k\} - \{\alpha_1, \dots, \alpha_q\}$.

Associated to the atom $R_{k+j}(\bar{z}_{k+j})$, we consider the formula denoted ϕ_i^j and defined as follows:

$$\phi_i^j = (\bar{z}_{k+j} \neq \bar{z}_{\alpha_1}) \wedge \dots \wedge (\bar{z}_{k+j} \neq \bar{z}_{\alpha_q})$$

where the expression $(\bar{z}_l \neq \bar{z}_s)$ denotes the following disjunction: $(t_l^1 \neq t_s^1) \vee \dots \vee (t_l^r \neq t_s^r)$, with $\bar{z}_l = (t_l^1, \dots, t_l^r)$, $\bar{z}_s = (t_s^1, \dots, t_s^r)$. In case when R_{k+j} does not occur in the positive part of f_i , then we consider $\phi_i^j = \text{TRUE}$. The formula $\phi(f_i)$ is defined as the conjunction of all formulas ϕ_i^j ,

for $1 \leq j \leq p$, that means $\phi(f_i) = \phi_i^1 \wedge \dots \wedge \phi_i^p$. Now, let us consider the conjunction of all formulas $\phi(f_i)$, $1 \leq i \leq m$, denoted $\phi(f_V)$, that means: $\phi(f_V) = \phi(f_1) \wedge \dots \wedge \phi(f_m)$. Let us consider an example concerning these formulas.

Example 2 Let V and I be defined in Example 1. To be short, let us rewrite the predicates $CON, PROD, ITEM$ by R_1, R_2, R_3 , respectively. We have: $V(x_1, x_2, x_3) : - R_1(x_1, x_2, y_1), R_1(y_2, y_3, x_2), R_2(x_3, x_4), R_3(x_1, x_3), \neg R_3(y_2, x_3)$. $f_1 = f_V(\bar{w}_1, \bar{y}) = R_1(1, S_2', y_1), R_1(y_2, y_3, S_2'), R_2(P_2', y_4), R_3(1, P_2'), \neg R_3(y_2, P_2')$. The formula $\phi(f_1)$ corresponds to the atom $R_3(y_2, P_2')$ and $\phi(f_1) = (y_2 \neq 1)$. For the formula f_2 we take y -variables as: y_5, y_6, y_7, y_8 . We obtain $\phi(f_2) = (y_6 \neq 2)$. Finally, $\phi(f_V) = (y_2 \neq 1) \wedge (y_6 \neq 2)$.

We remark that the formula $\phi(f_V)$ express the satisfiability property of the formula f_V . In the following we define formally the logic value of a formula for a C -partition.

Definition 2: Let π be a C -partition defined on $C \cup Y$ and $\phi(f_V)$ the formula constructed for f_V , as we have mentioned above. We define the logic value of $\phi(f_V)$ for π , denoted $\pi(\phi(f_V))$, as follows:

- (i) If $\phi = (t \neq t')$, where t and $t' \in C \cup Y$, then $\pi(\phi) = \text{TRUE}$ if there is no class E from $Class_\pi$ such that $t, t' \in E$, i.e., $[t]_\pi \neq [t']_\pi$.
- (ii) $\pi(\phi_1 \wedge \phi_2) = \pi(\phi_1) \wedge \pi(\phi_2)$, $\pi(\phi_1 \vee \phi_2) = \pi(\phi_1) \vee \pi(\phi_2)$.

We remark that for a C -partition π , we have $\pi(\phi(f_V)) = \text{TRUE}$ if and only if $\varphi_\pi(f_i(\bar{w}_i, \bar{y}_i))$ is satisfiable ([12]) for each $i, 1 \leq i \leq m$, where φ_π is the canonical onto mapping corresponding to π . For a database D defined on Dom , we consider $val(D)$ the set of all values that occur in the atoms of D . Formally,

$$val(D) = \{v | \exists R(\bar{w}) \in D, v \text{ is a component of } \bar{w}\}.$$

Let us denote by $f_1 \cdot f_2$ the composition of the mappings f_1 and f_2 , where $(f_1 \cdot f_2)(x) = f_1(f_2(x))$.

IV. A REPRESENTATION OF CERTAIN ANSWER SETS UNDER OWA

Firstly, we point out a proposition about two databases that are in a particular relation.

Proposition 1: Let D' and D be two databases over the schema S such that $D' \subseteq D$ and for each atom $R(\bar{w}) \in D - D'$, there exists a component v belonging to \bar{w} such that $v \notin val(D')$. Then for each query Q having $Rel(Q) \subseteq Rel(V)$, we have $Q(D') \subseteq Q(D)$.

Proof: Let \bar{u} be from $Q(D')$. This implies there exists a substitution θ from the variables of Q into Dom such that:

$$\theta pos(Q) \subseteq D', \theta neg(Q) \cap D' = \emptyset \text{ and } \theta \bar{z} = \bar{u}, \quad (5)$$

where the head of the query Q is $q(\bar{z})$. The hypothesis, the second statement from (5), and the safeness property of negation imply $\theta neg(Q) \cap D = \emptyset$, hence $\bar{u} \in Q(D)$. ■

The following theorem points out some properties concerning the C -partitions from $Part(C \cup Y)$ and some sous-databases

of a database D .

Theorem 1: Let \mathcal{V} be a set of view definitions on the schema \mathcal{S} , I an extension of \mathcal{V} , and D a database on Dom such that $I \subseteq \mathcal{V}(D)$, and $Rel(D) \subseteq Rel(\mathcal{V})$. Let $\phi(f_V)$ be the formula constructed for I and \mathcal{V} . There exist a C -partition π from $Part(C \cup Y)$ such that $\pi(\phi(f_V)) = TRUE$, a database D' such that $D' \subseteq D$, and a bijective mapping ψ_π from $Class_\pi$ into $val(D')$ having the following properties:

- (i) For each atom $R(\bar{w})$ from $D - D'$, the vector \bar{w} has at least a component t such that $t \notin val(D')$.
- (ii) For any query Q such that $Rel(Q) \subseteq Rel(\mathcal{V})$, we have: $Q(D') \subseteq Q(D)$ and $\psi_\pi^{-1}Q(D') = Q(\psi_\pi^{-1}(D'))$.
- (iii) Let $T = \psi_\pi^{-1}(D')$. We have $T \in \mathcal{M}_\pi$.
- (iv) $I \subseteq \mathcal{V}(D')$.

Proof: Let \mathcal{V}, I, D as in the hypothesis of the Theorem such that $I \subseteq \mathcal{V}(D)$. This inequality is equivalent to the statement: $V(\bar{w}_i) \in \mathcal{V}(D)$ for each $i, 1 \leq i \leq m$. This means:

$$(\exists \tau_i)(\tau_i : C \cup \bar{y}_i \rightarrow Dom)[D \models \tau_i f(\bar{w}_i, \bar{y}_i)], 1 \leq i \leq m \quad (6)$$

Moreover, we assumed that $\tau_i(c) = c$ for each element c from C . Let us emphasize the atoms from f_i :

$$f(\bar{w}_i, \bar{y}_i) = A_1 \wedge \dots \wedge A_h \wedge \neg A_{h+1} \wedge \dots \neg A_{h+p} \quad (7)$$

The relation $D \models \tau_i f(\bar{w}_i, \bar{y}_i)$ is equivalent to:

$$\tau_i A_j \in D, 1 \leq j \leq h \text{ and } \tau_i A_{h+l} \notin D, 1 \leq l \leq p \quad (8)$$

Since for $i \neq j$ we have $\bar{y}_i \cap \bar{y}_j = \emptyset$, there exists a mapping τ from $C \cup Y$ into Dom such that $\tau(c) = c$ for each c from C , and $\tau(y_\alpha) = \tau_i(y_\alpha)$, where $y_\alpha \in \bar{y}_i$. Associated to the mapping τ , we define a partition denoted π , and defined as follows: $t \equiv_\pi t'$ if $\tau(t) = \tau(t')$, where $t, t' \in C \cup Y$. Since the statements from (6) are true, it follows that $\pi(\phi(f_V)) = TRUE$. Let V' be the set of all values from $\tau(C \cup Y)$. Let D' be the database defined as follows:

$$D' = \{R(\bar{w}) | R(\bar{w}) \in D, R \in Rel(V) \text{ and}$$

$$\bar{w} \text{ contains only values from } V'\} \quad (9)$$

It is clear that $val(D') = V'$, and the databases D, D' satisfy the statement (i) from the Theorem. Using this statement and Proposition 1, we obtain $Q(D') \subseteq Q(D)$ for each query Q .

Now, let us define a bijective mapping denoted ψ_π from $Class_\pi$ into $val(D')$, as follows; $\psi_\pi([t]_\pi) = \tau(t)$. Let us denote by ψ_π^{-1} the inverse mapping of ψ_π . Let us show the second part of the condition (ii). Let Q be a query defined on \mathcal{S} and, having the form:

$$Q : q(\bar{z}) : -S_1(\bar{w}_1), \dots, S_l(\bar{w}_l), \neg S_{l+1}(\bar{w}_{l+1}), \dots, \\ \neg S_{l+r}(\bar{w}_{l+r}) \quad (10)$$

Let $q(\bar{w})$ be from $Q(D')$. There exists a substitution θ from the set of all variables from Q into V' such that the following statements yield:

$$\theta S_i(\bar{w}_i) \in D', 1 \leq i \leq l, \theta S_{l+i}(\bar{w}_{l+i}) \notin D', 1 \leq i \leq r,$$

$$\theta(\bar{z}) = \bar{w} \quad (11)$$

Since the mapping ψ_π^{-1} is injective, from the relation (11), we get:

$$\begin{aligned} \psi_\pi^{-1}(\theta S_i(\bar{w}_i)) &\in \psi_\pi^{-1}(D'), 1 \leq i \leq l, \psi_\pi^{-1}(\theta S_{l+i}(\bar{w}_{l+i})) \\ &\notin \psi_\pi^{-1}(D'), 1 \leq i \leq r, \psi_\pi^{-1}(\theta(\bar{z})) = \psi_\pi^{-1}(\bar{w}) \end{aligned} \quad (12)$$

From the relations (12), we infer the substitution $\theta' = \theta \cdot \psi_\pi^{-1}$ satisfies the relations (11) with θ' instead of θ and $\psi_\pi^{-1}(D')$ instead of D' . This means the following statement is true:

$$\psi_\pi^{-1}(q(\bar{w})) \in Q(\varphi_\pi^{-1}(D')) \quad (13)$$

$$\text{The relation (13) implies } \psi_\pi^{-1}(Q(D')) \subseteq Q(\psi_\pi^{-1}(D')) \quad (14)$$

The inclusion $Q(\psi_\pi^{-1}(D')) \subseteq \psi_\pi^{-1}(Q(D'))$ follows in a similar manner, because ψ_π^{-1} is bijective. Now, let us consider the statement (iii). Let $T = \psi_\pi^{-1}(D')$. Since the mapping τ satisfies the relation $\tau(\bigcup_{i=1}^{i=m} pos(f(\bar{w}_i, \bar{y}_i))) \subseteq D'$, we obtain the following inclusion:

$$(\tau \cdot \psi_\pi^{-1})(\bigcup_{i=1}^{i=m} pos(f(\bar{w}_i, \bar{y}_i))) \subseteq \psi_\pi^{-1}(D') \quad (15)$$

On the other hand, we get $\tau \cdot \psi_\pi^{-1} = \varphi_\pi$.

$$\text{Therefore, from (15), we obtain: } T_\pi^{min} \subseteq T \quad (16)$$

Since the relation $\tau(\bigcup_{i=1}^{i=m} neg(f(\bar{w}_i, \bar{y}_i))) \cap D' = \emptyset$ holds, we obtain:

$$\varphi_\pi(\bigcup_{i=1}^{i=m} neg(f(\bar{w}_i, \bar{y}_i))) \cap \psi_\pi^{-1}(D') = \emptyset \quad (17)$$

Moreover, we have:

$$\psi_\pi^{-1}(D') \subseteq \{\varphi_\pi R(\bar{w}) | R \in Rel(V), \bar{w} \text{ is on } C \cup Y\} \quad (18)$$

$$\text{The relations (17) and (18) imply } \psi_\pi^{-1}(D') \subseteq T_\pi^{max} \quad (19)$$

From the statements (16) and (19), we obtain $T \in \mathcal{M}_\pi$. The statement (iv) results because the relation (6) is satisfied for the database D' . ■

The following theorem specifies the properties of C -partitions.

Theorem 2: Let π be a C -partition from $Part(C \cup Y)$ such that $\pi(\phi(f_V)) = TRUE$. Let T be an element from \mathcal{M}_π . For each injective C -mapping ψ from $Class_\pi$ into Dom , we have:

$$(i) I \subseteq \mathcal{V}(D'), \text{ where } D' = \psi(T).$$

Proof: We consider the substitution τ from $C \cup Y$ into Dom , defined as: $\tau = \varphi_\pi \cdot \psi$. Let τ_i be the substitution obtained from τ by projection on $C \cup \bar{y}_i$, $1 \leq i \leq m$. We must show that:

$$D' \models \tau_i f(\bar{w}_i, \bar{y}_i), \text{ for each } i, 1 \leq i \leq m \quad (20)$$

Since T is a database from \mathcal{M}_π , we get for each $i, 1 \leq i \leq m$:

$$\varphi_\pi pos(f(\bar{w}_i, \bar{y}_i)) \subseteq T \text{ and } \varphi_\pi neg(f(\bar{w}_i, \bar{y}_i)) \cap T = \emptyset \quad (21)$$

Applying the mapping ψ to the first relation from (21), we obtain:

$$\tau pos(f(\bar{w}_i, \bar{y}_i)) \subseteq D', \text{ hence } \tau_i pos(f(\bar{w}_i, \bar{y}_i)) \subseteq D' \quad (22)$$

Since the mapping ψ is injective, from the second relation from (21), we get:

$\tau_{neg}(f(\bar{w}_i, \bar{y}_i)) \cap D' = \emptyset$, hence we have:

$$\tau_i neg(f(\bar{w}_i, \bar{y}_i)) \cap D' = \emptyset \quad (23)$$

The statements (22) and (23) imply (20), therefore we have $I \subseteq \mathcal{V}(D')$. ■

In the following we emphasize other property of C -partitions and injective mappings.

Proposition 2: Let π be a C -partition on $C \cup Y$ such that $\pi(\phi(f_V)) = \text{TRUE}$. Let D_1 be a database defined on $Class_\pi$, and ψ an injective mapping from $Class_\pi$ into Dom . We have $\psi(Q(D_1)) = Q(\psi(D_1))$, for each query Q expressed as a union of conjunctive form, and having $Rel(Q) \subseteq Rel(V)$.

Proof: Let Q be a query having the form like as in (10), and ψ an injective mapping from $Class_\pi$ into Dom . The answer of Q for T is as follows:

$$Q(T) = \{\theta q(\bar{z}) | \theta pos(Q) \subseteq T \text{ and } \theta neg(Q) \cap T = \emptyset\} \quad (24)$$

From this relation, we get:

$$\begin{aligned} \psi(Q(T)) &= \{(\theta \cdot \psi)q(\bar{z}) | \theta pos(Q) \subseteq T \text{ and} \\ &\quad \theta neg(Q) \cap T = \emptyset\} \end{aligned} \quad (25)$$

Let \tilde{u} be from $\psi(Q(T))$. There exists θ a mapping from the variables of Q into $Class_\pi$ such that $\tilde{u} = (\theta \cdot \psi)q(\bar{z})$, and θ satisfies the statements from (25). Since ψ is injective, from these relations, we obtain $(\theta \cdot \psi)pos(Q) \subseteq \psi(T)$ and $(\theta \cdot \psi)neg(Q) \cap \psi(T) = \emptyset$. These imply: $\tilde{u} = (\theta \cdot \psi)q(\bar{z}) \in Q(\psi(T))$. Therefore, we have obtained $\psi(Q(T)) \subseteq Q(\psi(T))$. The inverse inclusion is inferred similarly. ■

Before we give the theorem about a representation of certain answers, we need to give some further notations. Let $P = \{\pi_1, \dots, \pi_p\}$ be the set of all C -partitions from $Part(C \cup Y)$ such that $\pi_i(\phi(f_V)) = \text{TRUE}$. For a partition π_i , we denote by S_{π_i} the intersection of all answers of the query Q for databases T from M_{π_i} , that means: $S_{\pi_i} = \cap\{Q(T) | T \in M_{\pi_i}\}$. Let $\mathcal{A} = (S_{\pi_1}, \dots, S_{\pi_p})$. Let ψ_i be an injective mapping from $Class_{\pi_i}$ into Dom , $1 \leq i \leq p$. Let \mathcal{B} be the vector (ψ_1, \dots, ψ_p) and $Ans(\mathcal{B}) = \cap_{i=1}^{i=p} \psi_i(S_{\pi_i})$. Let $VMap$ be the set of all vectors having the form \mathcal{B} , and $RCertAnsO$ the intersection of all $Ans(\mathcal{B})$ for all \mathcal{B} from $VMap$, i.e., $RCertAnsO = \cap\{Ans(\mathcal{B}) | \mathcal{B} \in VMap\}$. Let us denote by $CertAnsO(\mathcal{V}, I, Q)$, the set of all certain answers for \mathcal{V}, I, Q . In the following theorem, we give a characterization of this certain answer set.

Theorem 3: Let \mathcal{V}, I, Q be a set of view definitions, an instance of \mathcal{V} and a query, respectively. We have $CertAnsO(\mathcal{V}, I, Q) = RCertAnsO$.

Proof: Firstly, let \bar{w} be a vector from $CertAnsO(\mathcal{V}, I, Q)$. To show that $\bar{w} \in RCertAnsO$. Let \mathcal{B} be a vector of injective mappings, $\mathcal{B} = (\psi_1, \dots, \psi_p)$, where ψ_i is a mapping from $Class_{\pi_i}$ into Dom . Let T be an element from M_{π_i} . Let us denote the database $\psi_i(T)$ by D'_i .

By Theorem 2, we have $I \subseteq \mathcal{V}(D'_i)$. Using the hypothesis, we obtain $\bar{w} \in Q(D'_i) = Q(\psi_i(T)) = \psi_i(Q(T))$. We get $\bar{w} \in \psi_i(S_{\pi_i})$, for each π_i from P and mapping vector \mathcal{B} , therefore $\bar{w} \in RCertAnsO$.

Inversely, assume that $\bar{w} \in RCertAnsO$. To show that $\bar{w} \in CertAnsO(\mathcal{V}, I, Q)$. Let D be a database on Dom such that $I \subseteq \mathcal{V}(D)$. We must show that $\bar{w} \in Q(D)$. By the hypothesis, we have $\bar{w} \in Ans(\mathcal{B})$, for each \mathcal{B} from $VMap$, hence we obtain:

$$\bar{w} \in \psi_i(S_{\pi_i}), \text{ for each } i, 1 \leq i \leq p, \text{ and for each } \mathcal{B}. \quad (26)$$

Using Proposition 2, we get: $\bar{w} \in \psi_i(Q(T))$, for each $T \in M_{\pi_i}$. Using Theorem 1, we have: there exist a partition π_i , a mapping ψ_{π_i} , and a database D' such that $D' \subseteq D$, $I \subseteq \mathcal{V}(D')$, where ψ_{π_i} is a mapping from $Class_{\pi_i}$ into $val(D')$. In the relation (26) we take ψ_{π_i} , instead of ψ_i . Thus, using Proposition 2, we have: $\bar{w} \in \psi_{\pi_i}(Q(T)) = Q(\psi_{\pi_i}(T)) = Q(\psi_{\pi_i}(\psi_{\pi_i}^{-1}(D')) = Q(D')$. Since $Q(D') \subseteq Q(D)$, we obtain $\bar{w} \in Q(D)$. ■

V. CERTAIN ANSWERS UNDER OWA

Based on the results of the precedence section, we give in this section a method to construct the set of all certain answers for \mathcal{V}, I, Q . Since we considered each constant from C belongs to the domain Dom , we have $C \subseteq Dom$. Moreover, for each π a C -partition on $C \cup Y$, we have $|C| \leq |Class_\pi|$, where $|C|$ denote the cardinality of C . Regarding to the vectors having components from $Class_\pi$, we need to introduce a condition denoted *Cond* and defined in the following.

Definition 3: Let $\tilde{w} = (\bar{w}_1, \dots, \bar{w}_p)$, where $\bar{w}_j \in S_{\pi_j}$, and $\bar{w}_j = (t_{j1}, \dots, t_{jr})$, $1 \leq j \leq p$. We say that the vector \tilde{w} satisfies the condition *Cond* if the following statements yield:

- (i) The class t_{ji} contains a constant denoted $c_{\alpha_{ji}}$ for all $j, 1 \leq j \leq p$, and $i, 1 \leq i \leq r$,
- (ii) Let $t_{jl} = [c_{\alpha_{jl}}]_{\pi_l}$, $1 \leq l \leq r, 1 \leq j \leq p$. Then we have: $c_{\alpha_{1l}} = \dots = c_{\alpha_{pl}}$, for each $l, 1 \leq l \leq r$.

In the following, we point out some properties of the vector \tilde{w} that satisfy the condition *Cond* from Definition 3.

Proposition 3: Let $\tilde{w} = (\bar{w}_1, \dots, \bar{w}_p)$, where $\bar{w}_j \in S_{\pi_j}$, $1 \leq j \leq p$. We have: the vector \tilde{w} satisfies the condition *Cond* if and only if there exists a unique injective C -mapping from $Class_{\pi_j}$ into Dom , denoted ψ_j , for each $j, 1 \leq j \leq p$ such that $\psi_1(\bar{w}_1) = \dots = \psi_p(\bar{w}_p)$.

Remark 1: Let $\tilde{w} = (\bar{w}_1, \dots, \bar{w}_p)$, where $\bar{w}_j \in S_{\pi_j}$, $1 \leq j \leq p$. The vector \tilde{w} produces a certain answer, denoted $PROD(\tilde{w})$, under OWA if and only if \tilde{w} satisfies the condition *Cond*. In this case, we have from Proposition 3, $PROD(\tilde{w}) = \psi_1(\bar{w}_1)$.

We can easy construct a procedure that computes the set of all certain answers for \mathcal{V}, I, Q .

Example 3 Let us consider I, V and Q as in Example 1. Using the results of Sections IV and V, we obtain that $w = (1)$ and $w = (2)$ are certain answers under OWA.

VI. CERTAIN ANSWERS UNDER CWA

In this section, we point out some theorems necessary to represent sets of certain answers under *CWA*. The proofs of these theorems are similar to that of Theorems 1, 2, 3, therefore they are omitted. Firstly, we need to consider a new notion regarding to a database T defined on $Class_{\pi}$, where π is a C -partition.

Definition 4: Let $I = \{\bar{w}_1, \dots, \bar{w}_m\}$, π a C -partition such that $\pi(\phi(f_V)) = \text{TRUE}$ and T an element from \mathcal{M}_{π} . We say that T is closed with respect to I , if for each substitution θ from the variable set of Q into $Class_{\pi}$ such that $T \models \theta f_V(\bar{x}, \bar{z})$, there exists a tuple \bar{w}_j from I such that $\eta(\theta(\bar{x})) = \bar{w}_j$, where η is the mapping from $Class_{\pi}$ into $C \cup Y$ defined by: $\eta([t]_{\pi}) = c$ if $c \in [t]_{\pi}$ and $\eta([t]_{\pi}) = y$ otherwise, where y is a variable from the class $[t]_{\pi}$.

Remark 2: Let I , π and T as in Definition 4. We have:

- (i) For each $i, 1 \leq i \leq m$, there exists a substitution θ_i from the variable set of Q into $Class_{\pi}$ such that $T \models \theta_i f_V(\bar{x}, \bar{z})$ and $\eta(\theta_i(\bar{x})) = \bar{w}_i$, where the mapping θ_i is specified in Definition 4.

Proof: We specify the substitution θ_i . If $\bar{x} = x_1 \dots x_h$, $\bar{z} = z_1 \dots z_p$, $\bar{w}_i = t_1 \dots t_h$, where $t_j \in Dom$, $\bar{y}_i = y_{\gamma_1} \dots y_{\gamma_p}$ (the vector \bar{y}_i consists of the y -variables from the expression $f_V(\bar{w}_i, \bar{y}_i)$). The mapping θ_i is defined as follows: $\theta_i(x_j) = [t_j]_{\pi}$, $1 \leq j \leq h$ and $\theta_i(z_j) = [y_{\gamma_j}]_{\pi}$, $1 \leq j \leq p$. ■
Now, we give the results regarding the representation of certain answer sets under *CWA*.

Theorem 4: Let \mathcal{V} be a set of view definitions, I an extension of \mathcal{V} , and D a database on Dom such that $I = \mathcal{V}(D)$. Let $\phi(f_V)$ be the formula constructed for I and \mathcal{V} . There exist a C -partition π from $Part(C \cup Y)$ such that $\pi(\phi(f_V)) = \text{TRUE}$, a database $D' \subseteq D$, and an injective mapping ψ_{π} from $Class_{\pi}$ into $val(D')$ having the following properties:

- (i) and (ii) as in Theorem 1,
- (iii)' Let $T = \psi_{\pi}^{-1}(D')$. We have $T \in \mathcal{M}_{\pi}$, and T is closed with respect to I (Definition 4).
- (iv)' $I = \mathcal{V}(D')$.

Theorem 5: Let π be a C -partition from $Part(C \cup Y)$ such that $\pi(\phi(f_V)) = \text{TRUE}$. Let T be a database from \mathcal{M}_{π} that is closed with respect to I . Then for each injective C -mapping ψ from $Class_{\pi}$ into Dom , we have:

- (i) $I = \mathcal{V}(D')$, where $D' = \psi(T)$.

Now, we use the notations specified for the case *OWA*, with except the following: \bar{S}_{π_i} instead of S_{π_i} , $CertAnsC(\mathcal{V}, I, Q)$ instead of $CertAnsO(\mathcal{V}, I, Q)$, and $RCertAnsC$ instead of $RCertAnsO$, where

$$\bar{S}_{\pi_i} = \cap\{Q(T) | T \in \mathcal{M}_{\pi_i} \text{ and } T \text{ is closed w.r.t. } I\}.$$

Theorem 6: Using the notations specified above, we have $CertAnsC(\mathcal{V}, I, Q) = RCertAnsC$.

VII. TIME COMPLEXITY TO COMPUTE CERTAIN ANSWERS

It is known that the total number of partitions of an n -element set is the Bell number B_n , such that the following recursion equation yields ([13]): $B_{n+1} = \sum_{k=0}^{k=n} B_k$. Using the induction, we get the following inequalities: $B_n > 2^n$ for each $n \geq 5$ and $B_n < n^n$ for each $n > 1$. The second inequality implies $B_n < 2^{n^2}$, for each $n > 1$. On the other hand, the number of C -partitions defined on $C \cup Y$ is greater than the number of the partitions on Y . It results that the number of C -partitions defined on $C \cup Y$ is of the type $O(2^{p(|Y|)})$, where p is a polynomial. Let us discuss the number of all elements from \mathcal{M}_{π} , where π is a fixed C -partition on $C \cup Y$. Let $y_{\pi} = |Class_{\pi}|$, and r the maximum of the arities of the relational symbols from V . Then the cardinality of the set T_{π}^{max} (in relation (4)) has the form: $O(y_{\pi}^r)$. Therefore, the number of the elements from \mathcal{M}_{π} has the form $O(2^{y_{\pi}^r})$. It is clear that $y_{\pi} \leq |C \cup Y|$. For a query Q having 1 variables, and T an element from \mathcal{M}_{π} , the number of the substitutions from $Var(Q)$ into $Class_{\pi}$ is y_{π}^l . Using these relations, and the constructions for certain answers under *OWA* and *CWA*, specified in Sections V and VI, we obtain that the time complexity to compute these certain answers is *EXPTIME*.

VIII. CONCLUSION

We have presented a representation of certain answers corresponding to a set of view definitions, a set of extensions of the view definitions, and a query. Two situations were considered: open-world assumption and close-world assumption. Using this representation, a method to compute the certain answers under the two assumptions was given.

REFERENCES

- [1] A. Y. Halevy, *Answering Queries Using Views: A survey*, VLDB Journal, vol. 10, nr. 4, 2001, pp. 270-294.
- [2] S. Abiteboul and O. M. Duschka, *Complexity of answering queries using materialized views*, in PODS, 1998, pp.254-263.
- [3] S. Flesca and S. Greco, *Rewriting queries using views*, IEEE Trans. Knowledge Data Engineering 13(6), 2001, pp.980-995.
- [4] D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi, *View-based query processing:On the relationship between rewriting, answering and losslessness*, Theoretical Computer Science, 371, 2007, pp. 169-182.
- [5] D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi, *Answering Regular Path Queries Views*, Proc. of the 16th IEEE Int. Conf. on Data Engineering, ICDE, 2000, pp. 389-398.
- [6] G. Grahame, and A. O. Mendelson, *Tableau technique for querying information sources through global schemas*, Proc. of the 7th Int. Conf. on Database Theory, ICDT'99, in LNCS, vol. 1540, Springer 1999, pp. 332-347.
- [7] A. Call, D. Lembro, and R. Rosati, *Query rewriting and answering under constraints in data integration systems*, IJCAI-03, 2003, pp.16-21.
- [8] F. Afrati, C. Li, and P. Mitra, *Rewriting queries using views in the presence of arithmetic comparisons*, Theoretical Computer Science, 368, 2006, pp.88-123.
- [9] O. M. Duschka, M. R. Genesereth, and A. Levy, *Recursive Query Plans for Data Integration*, J. of Logic Programming 2000, 43(1), pp. 49-73.
- [10] T. Millstein, A. Halevy, and M. Friedman, *Query containment for data integration systems*, JCSS vol.66, 2003, pp. 20-39.
- [11] Wei F. and Lausen G.: *Containment of Conjunctive Queries with Safe Negation*. In ICDT, (2003), LNCS, vol.2572, 346-360.
- [12] J. Hammer, H. Garcia-Molina, S. Nestorov, R. Yerneni, M. M. Breunig, and V. Vassalos, *Template-based wrappers in the TSIMMIS system*, Proc. of ACM SIGMOD, 1997, pp. 532-535.
- [13] R.A.Bruazzi, *Introductory Combinatorics* (4th edition ed.), Pearson Prentice Hall, ISBN 0131001191, 2004.

Formation of Triads in Mobile Telecom Networks

N Naren Krishna
Ericsson India Private Limited
Ericsson R&D, Chennai, India
naren.krishna@ericsson.com

M Saravanan
Ericsson India Private Limited
Ericsson R&D, Chennai, India
saravanan.r.mohan@ericsson.com

Abstract— In the present competitive telecom scenario, an intention of any operator is to increase the size of the network and establish more connectivity between its users. This can be achieved either by adding new customers to the network or by increasing number of links in the network. In this paper, we present a method called triad formation for increasing the connectivity in the network. Community detection has been done on the network before to set up the triads for efficient results. The communities formed are based on the modularity factor. The proposed method will resolve possible new combination of edges between nodes which are not connected earlier and it has a strong connection with a common node. The effectiveness of the triad formation is demonstrated on a huge telecom data and its importance is highlighted.

Keywords: Call Detail Record (CDR); Modularity; Isolated Community; Triads; Mobile Social Network Analysis(MSNA).

I. INTRODUCTION

A social network is a compact structure which explores connected groups and it is used to predict the actions of individuals. These individuals in a network are called nodes and their communications are measured in the form of dependent edges. Dependency varies from friendship, common interests, beliefs and knowledge. One of the utmost interests of telecom operators is to increase the connectivity between customers to generate more profit. For this, they need to identify the existence of strong connections and influential members based on the mobile usage services. Mobile Social Network Analysis (MSNA) is an upcoming research area which indicates the importance of identifying the social groups in mobile networks [1]. It helps the operators in understanding and analysing the subscribers and increases the focus on their business. MSNA has proved to give extensive results in the areas like churn prediction, customer retention and campaign management [2].

The behaviour of highly connected customers and their relationship mainly depends on the social structure of the communities. With the help of social network analysis it is possible to find the potential users, influential members and weak users of a community [3]. The real challenge in telecom networks is its dimension. Processing millions of nodes and billions of edges is not only a tough task, but also a time taking process. One of the solutions to this challenge would be detecting communities and then performing analysis on individual communities which will derive efficient results. Community detection can be done based on common interests of the subscribers and their calling patterns. There are several interesting algorithms for community detection available which can segment the network based on modularity [3, 4, 5]. Modularity is a prime factor which defines the strength of a community. CNM

algorithm is one of its kind which helps in finding communities using a hierarchical agglomeration algorithm for detecting community structure which is faster than many competing algorithms [4]. CNM algorithm works on undirected edges in a network. Hence, whenever it runs through a cycle in the network, the algorithm runs into an infinite loop and leads to a memory constraint and stops working. The algorithm has been slightly modified to solve this problem.

We propose a novel idea on new group formation called triads, which finds nodes which were not connected earlier and where there is a high probability of forming new edges between them because they have a strong common node. By doing this, the number of edges (number of calls) increase which in turn leads to increase in revenue for the operator. To the best of our knowledge, this is the first attempt to generate new edges in a telecom network at a community level. The main contribution of our research is based on the telecom data obtained from the Call Data Records (CDRs) which are used for community detection and formation of triads. The communities which are formed with higher modularity help the operator to understand its customers as a group and it is easy to analyse the behaviour of potential customers.

The first and foremost step is to clean the data to avoid the redundant and unnecessary attributes. The attributes of CDRs which are taken into consideration are Calling Number, Called Number and Duration of the call. This cleansed data will be used as input for the modified CNM algorithm which will detect communities. Once the communities with high modularity are detected, the task of finding nodes which have high probability in making new edges is easy which will lead to the formation of triadic closure [5] in mobile networks. A brief stepwise description of our research work is as follows

A. Data Pre-processing

First and foremost step is to clean data or in other terms filter out those set of attributes from CDRs which are not used in the next step of our research. In a telecom scenario for every call made, a CDR is generated. Records related to the calls made to toll free numbers and call centres should be filtered out. Records indicating voice calls are collected. Our study on triadic formation is based only on voice calls.

B. Classification of users

To enrich the results and to achieve a better accuracy, it is suggested to classify users into weak and potential users. It is a challenging task for an operator to find out its active or potential users who add weight to their network. Such classification rules would solve the problem and does make

it easy to identify related users and in order helps to understand their behaviour and also the behaviour of the community the user is related to. It is at the ease of operator to customize the rules and classify the users based on his need.

C. Community Detection

Communities are formed based on the increase or gain in the modularity when nodes merge [6]. This is a recursive step and stops when there is no more gain in the modularity. Unlike other greedy algorithms, data structures have been used to handle sparse adjacency matrices.

D. Identifying Isolated Communities

Communities of a network which do not have any edge or connection to other communities are called isolated communities. The users of these communities do not have any edges or connections to other community members. The total number of inter-community calls is zero. Communities which have more than one edge or connection to other communities are called non-isolated communities. The need for identifying isolated communities drills down the problem of understanding the customers, as the behaviour of an isolated community depends on the behaviour of the alpha user of that community. This is the major advantage of isolated communities. An alpha user is a highly potential node who influences the community [1] and the community behaviour can be concluded based on the alpha user's behaviour and the time taken to spread the information to the network from this point is minimal. Another major advantage of an isolated community is that the community does not get disturbed by external factors.

E. Generation of New Edges

This is the final step of our research work. Generating new edges between nodes which are not connected and which have a strong common node.

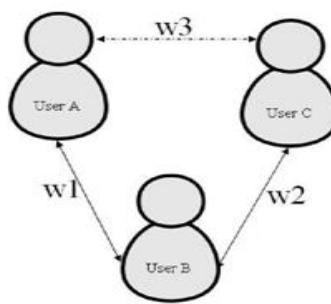


Figure 1 Triad formation

This step is called triad formation which is represented in Fig. 1. It shows three users A, B and C where User B is the common node. The edge AB whose edge weight is represented by w1 and edge BC whose edge weight is represented by w2 are the existing edges and the dotted edge

AC denote the newly generated edge. The edge weight w3 of edge AC is generated.

The rest of the paper is organized as follows. In Section II, we review the related literatures and in Section III, we describe the procedure of experiment at different stages. Section IV gives our experiment results followed by discussion in Section V and concluding remarks and future work in Section VI.

II. RELATED WORK

Understanding the customers and making them feel comfortable without creating any disruption in the network is considered to be the top most priority of any network provider. Mobile Social Network Analysis (MSNA) is an evolving research area where many challenges like finding influential members in network or community, detecting community of high modularity, understanding the behavior of a community, churn prediction are yet to find perfect solutions. In this section we describe some important studies which laid roots to community detection and triadic formation in mobile networks.

A. Stanley six degree experiment

Stanley Milgram is famously known for his "Small World Phenomenon" [7]. He routed messages in a network based on the familiarity of the first name and deduced to an interesting result which states that any person can be reached by any other person by not more than six hops. In simple terms, distant people can be connected by short paths in which every edge connects two people who know each other quite well. The data considered in this experiment was not as complex as the telecom data. This experiment has provoked us to experiment with mobile networks to explore the existence of knitted communities. Recent work has suggested that this phenomenon has effect in networks arising from nature and technology, and is a fundamental ingredient in the structural evolution of the World Wide Web. Such experiments on telecom network would give us an idea about the structure of the network. The number of hops or the time taken for information or message to exchange between any two random nodes can be deduced.

B. Pareto's 80/20 rule

It states that 20% of landowners own 80% of the land, 20% of the workers do 80% of the work, 20% of criminals carry out 80% of the crime and 20% of websites get 80% of the traffic. This rule when tuned to the telecom network it will give us an idea that 20% of our customers make 80% of our business [8]. Identifying these set of customers is one of the challenges as mentioned previously and these customers are called alpha users in many studies.

C. Community Detection

There have been many community detection algorithms seen in recent times. CNM algorithm [4] is one of the best of its kind. It is a hierarchical agglomerative algorithm which iteratively merges nodes into communities based on the gain in modularity. Its running time on a network with n vertices and m edges is $O(m d \log n)$ where d is the height

of the dendrogram. Fast unfolding [9] is another community detection algorithm based on modularity. It is divided into two phases where in the first phase every node has its community number. Then the modularity is calculated with respect to all its neighbors. If there is positive gain in the modularity then the nodes are merged.

The CNM algorithm at this point has to be modified accordingly in order to meet few exceptions. The algorithm is purely based on graph theory where it considers every user as a node. It does not work under the condition where a loop exists between two nodes. A loop in telecom network can be understood as a bi-directional edge between two users. In order to solve this problem the algorithm has been slightly modified to consider the directional edges. Whenever such situations are raised the source and destination are swapped in order to overcome the loop as the bidirectional edge is made into two unidirectional edges.

D. Triadic closure

Triadic closure is a concept in social network theory, first suggested by German sociologist Georg Simmel in the early 1990s. It is a property between 3 nodes A, B and C. Suppose there exists a strong tie between A-B and B-C, then there will be a weak or strong tie between A-C. In [5], Mark Granovetter has synthesized the theory called "Cognitive Balance" which refers to the tendency of two individuals who feel the same way about an object. If the triad between three nodes is not closed, then the node connected to both the nodes has the ability to close this triad in order to complete the closure in the relationship network.

III. PROCEDURE

A stepwise presentation of our research work has been presented in this section. The unnecessary records from CDRs are filtered out and for better results the users are classified into active and potential users. After the users are classified into the mentioned categories, they are used as input data for the community detection algorithm which results to isolated and non-isolated communities. The last step which is the major part of our research is the generation of new edges between nodes which are not connected and have a strong common node. The new edges which are generated complete the triad formation.

A. Data processing

The telecom data used here had 1.2 million CDRs from African Telecom operator. This data includes the CDRs related to voice, SMS, GRPS. The data used here was over duration of week. Our research is carried only on voice calls and hence the first step is to filter out all the call records related only to voice calls. The next step of data cleaning is to identify the called numbers which have an outdegree 0 or indegree 0. Outdegree 0 can be defined as the caller ID which in all cases is the destination of call and not a source. Indegree 0 can be defined as the caller ID which in all cases is the source of call and never a destination.

- Example of Outdegree 0: Toll Free numbers and call centers
- Example of Indegree 0: IVRS generated

advertisement.

As the caller ID's are nearly 10 digits in length, processing them at every step is yet another time consuming process. Unique ID's were given to the users to overcome this problem.

B. Classification of users

The filtered data from the processing step is used for this step. The attributes used are Calling Number, Called Number and Time Duration. The data has 1,82,000 distinct users.

Users have been classified into weak and potential users based on their calling patterns and their behavior in the network. Thresholds have been set on Indegree (number of calls received) and Outdegree (number of calls made). Certain rules have been followed in order to classify the users accordingly. These rules are user specific and depend on the data used. The rules designed by us are on the basis of the 7 days data on which we carried our experiments. The following are the rules laid down by us which classify a user as a potential user:

- 1) User has to be active on any 4 days of the week ,
- 2) Indegree of the user should be greater than or equal to 10,
- 3) Outdegree of the user should be greater than or equal to 10,
- 4) Sum of indegree and outdegree should be greater than or equal to 20.

The basis for the rules is as follows:

Rule 1) User has to be active on any 4 days of the week:

We found an interesting pattern on observing the data over the duration of the data. Some of the users were active only on the first day of the week, some are active on the last day of the week and most of the users were active for an average of 4 days of the week. This has been calculated by noting down the availability (A) and non-availability (NA) of the user over a week. The presence of a user j on a day i is defined as P_{ij} . The value of P_{ij} is 1 if the user is present on day i else it is assigned 0. The Average Active Period (AAP) is defined as

$$AAP = \frac{\sum_j^{\text{users}} \sum_i^{\text{days}} P_{ij}}{\text{total number of users}} \quad (1)$$

Rule 2) Indegree of the user should be greater than or equal to 10:

Fig. 2 plots the net indegree or number of calls received by every user over a week and average indegree of a user over a week turned out to be 10. The horizontal axis is indicated by the users of the network by their distinct ID's and the vertical axis has the Indegree of every user.

Rule 3) Outdegree of the user should be greater than or equal to 10:

Fig. 3 plots the net outdegree or number of calls made by

every user over a week and average outdegree of a user over a week turned out to be 10. The horizontal axis is indicated by the users of the network by their distinct ID's and the vertical axis has the Outdegree of every user.

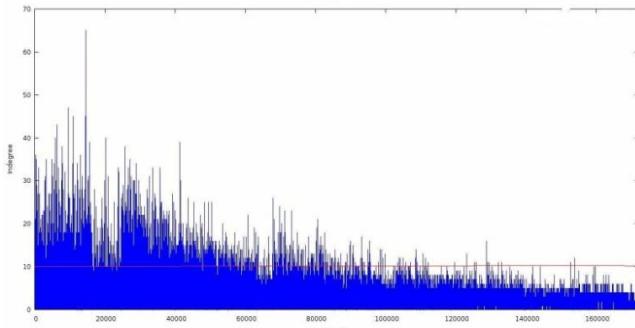


Figure 2 Indegree Vs User Id

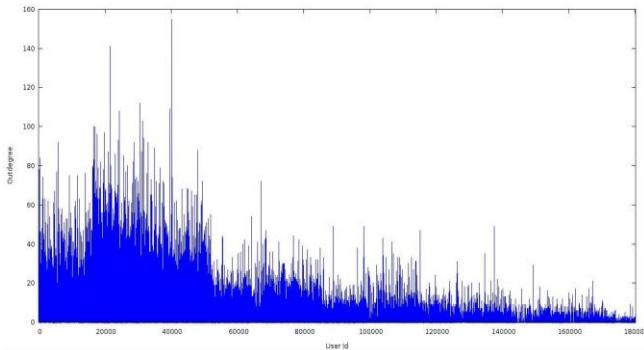


Figure 3 Outdegree Vs User Id

Rule 4) Sum of indegree and outdegree should be greater than or equal to 20:

This rule has the roots from rule 2 and 3. The sum of average indegree and average outdegree leads to 20. This rule has been made to ensure that we do not have users who just receive calls or make calls.

User who satisfies all the four rules is classified as *potential users* and the rest are called *weak users*. The latter are not considered as input set for further steps in the study. These users might increase the revenue of the network but the network or community does not shatter once they churn out and these set of users cannot be exactly put into a particular community. Addition of these users in the collection might lead into biased results. Following are the examples for weak users

1) *Call-Center*: The indegree of every call center is high when compared to its outdegree. It is impossible to categorize them into a particular community

2) *Business scheme promoters*: The indegree in this case is negligibly small when compare to its outdegree. These set of users are also considered as weak users.

At the end of this stage our data set consists of only active users. Calling number, called number and duration of

the call are the only attributes which are being used. At the beginning of this step we had 1,82,000 distinct users, and after classification only 68,000 users are classified as active.

C. Community Detection

Community detection has been the toughest challenge in research area in present days. CNM performs the same greedy optimization as the algorithm [10]. What makes this perform better than the existing one is the way it handles the sparse matrices. Sparse matrices are faced in the early stages of formation of adjacency matrices and this operation is efficiently carried out in CNM using data structures for sparse matrices.

The calling number and the called number are taken as inputs for the community detection algorithm. CNM algorithm [4] is used for community detection. It is a hierarchical agglomeration algorithm which detects the community structure and is faster than many competing algorithms. Initially every node is considered as a community and adjacency matrix A_{vw} is generated for that network. $\delta(c_v, c_w)$ denotes the connectivity. δ equals 1 if there exist an edge, else it is 0. The number of edges m in the graph and vertex x belonging to community c_v is given by

$$m = \frac{1}{2} \sum_{vw} A_{vw} \delta(c_v, c_w) \quad (2)$$

The fraction of edges making one community is given by

$$\frac{\sum_{v,w} A_{vw} \delta(c_v, c_w)}{\sum_{v,w} A_{vw}} = \frac{1}{2m} \sum_{vw} A_{vw} \delta(c_v, c_w) \quad (3)$$

Modularity is the main parameter which is taken into consideration when the communities are made. The fraction is directly proportional to the probability of the two edges merging into one community. Degree k_v of vertex v is defined as the number of edges leading to it, then the modularity (Q) is defined as

$$Q = \frac{1}{2m} \sum_{vw} \left[A_{vw} - \frac{k_v k_w}{2m} \right] \delta(c_v, c_w) \quad (4)$$

The input handling has been changed to avoid the algorithm from running into an infinite loop. Eqn. 5 shows that CNM algorithm [4] works only for undirected edges. Every edge in telecom network is directed. In other terms every edge in telecom network has a source (calling number) and destination (called number). CNM does not identify such edges and considers them as undirected edge which leads the algorithm into an infinite loop. The modification which is made into it was at the input stage for algorithm. The input file format is a list of called and calling numbers.

$$\delta(c_v, c_w) = \sum_i \delta(c_v, i) \delta(c_w, i) \quad (5)$$

The condition checkpoint which was introduced here is: If there exists an edge as A to B and B to A then swap B and A in the second case or else there would be an infinite loop between A to B and algorithm stops working. Once the nodes are swapped in the second case there will only be 2 copies of edges between A and B which resulted in dissection of loop which solves the problem. Upon every merge, the rows and columns of the corresponding communities are blended. Alternate way is to store the difference in the values of the modularity which result from the conglomeration of communities. Choose the largest among them and perform amalgamation. This would result in lesser memory storage and computational cost as these are the main hurdles when we are dealing with huge data.

The strongest user of a community is the alpha user of the community [1]. Definition of alpha user varies from research and its usage. In our research it is defined as the user of a community on whose removal there is a high probability of the network getting shattered off. The user is highly connected to most of the users in that community. The user has the capability to spread the information in less time.

D. Isolated and Non-Isolated Communities

Every community formed is given a unique ID. Every community is now considered as a node and the degree of the community is calculated. No of edges from a particular community to other community is defined as the degree. If the degree is equal to 0 then it is called an isolated community else a non-isolated community. Higher the number of isolated communities, higher is the modularity of the community as modularity is defined as the ratio of number of isolated communities to the total number of communities formed. Isolated communities are those whose members make calls only to their community members and make no calls to any other community members.

E. Generation of New edges

All the communities and the community members along with their respective CDRs which contain the calls made by them or call received by them along with the duration are taken as input. We have presented the algorithm related to the generation of new edges in mobile telecom networks.

Algorithm A graph G = (V, E) with V>0 and E>0

- 1) Look for two nodes x, y that are non-adjacent and share a common neighbor
- 2) If no such pair of nodes x, y exists, go to step 8, else goto step 3
- 3) The edge weight between x and the common neighbor is labeled as w1
- 4) The edge weight between y and the common neighbor is labeled as w2
- 5) Check if w1 and w2 are more than the threshold weight set by the user
- 6) If not go to step 8, else goto step 7
- 7) Add edge(x, y) and assign a weight equivalent to average of w1 and w2
- 8) Add element x, y to set E
- 9) Go to step 1.

The implementation of the algorithm is as follows:

Formation of new edges between nodes which are not connected and have a strong common node in between is the motto of the algorithm. To enhance our results, we will concentrate on every community individually and try to form new edges in between users of the community. These nodes can be easily joined because they belong to the same community. Certain rules have been laid down in computing such edges. Let's say there are three nodes A, B, and C as shown in Fig. 1, where AB and BC are connected and B is the common node. Weight of the edge AB is w1 and weight of the edge BC is w2. A new edge AC will be formed if and only if w1 and w2 are more than the threshold weight. This threshold weight is user specific. This is used to define whether the edges w1 and w2 (already existing edges) are strong or weak. Once this new edge is formed its weight is defined as the average of w1 and w2. Formation of new edges not only increase the weight of the structure but also increase the closeness, clustering co-efficient, average talk time and average revenue of the community. The betweenness values of the community decreases which actually indicates that the probability of the network getting disturbed when a node or user moves to other network is less.

IV. RESULTS

The section describes the results achieved. The result consists of communities deduced with high modularity over duration of seven days. Another experiment which was carried on was to find out the behavior of communities over two time periods, namely weekends and weekdays. The alpha members of every community have been highlighted using Pajek software [11] which is a tool used for network analysis and visualization. In the last section we have presented the results on triads using a community of size 339 nodes as network.

TABLE I. ATTRIBUTES OF COMMUNITY

Period	Modularity	No. of Communities	Min Size	Max Size	Mean size
Week (7 days)	0.9970	8533	2	281	3.52
Weekdays	0.9506	9613	2	624	5.48
Weekends	0.9704	9703	2	1086	6.87

Table I shows the behavior of communities that are formed over three time periods namely weekdays, weekends and over a week. It is shown that the communities that are formed on weekends have better modularity, mean size and max than when compared to the communities formed on weekdays. This is because the users in the network are active over the weekends. The above table shows attributes of communities that are formed over two time periods namely weekdays and weekends. Communities that are formed on weekends have a better modularity, mean size and max size than the communities formed on weekdays. This is because the number of calls and the duration of calls are high over the weekends.

Table II shows the number of communities and their respective degrees. As mentioned earlier a community with degree 0 is called an isolated community where the users belonging to that community only make intra-community calls.

Interesting results which were deduced by the formation of community over a week

1. 8510 out of 8533 communities are isolated
2. Every community has an alpha member. Hence there are 8533 alpha members in the network
3. Number of communities which have only one edge to other communities: 8

TABLE II. COMMUNITIES AND THEIR DEGREE FORMED OVER A WEEK

Degree	No. of Communities
0	8510
1	8
2	4
3	4
4	5
5	2

TABLE III. DISTRIBUTION OF ISOLATED COMMUNITIES FORMED OVER A WEEK

Community Size(size)	No. of Communities
1 < size < 10	8374
10 < size < 20	86
20 < size < 30	22
30 < size < 40	14
40 < size < 50	9
50 < size < 60	5
60 < size < 100	2

Table III describes the distribution of isolated communities over a week. Size of a community indicates the number of users in that particular community. 8374 communities are of size less than 10, which indicates that there are more number of smaller communities when compared to communities of size 100.

Fig. 4 represents the hierarchical representation of a community of users based on their net degree in ascending order from the top. This is a sample representation of one of the community whose size is 339. The node at the bottom of the diagram is the alpha member. It can be clearly seen that the number of connections between the alpha user to the other users in the community. The users at a particular level have equal degree.

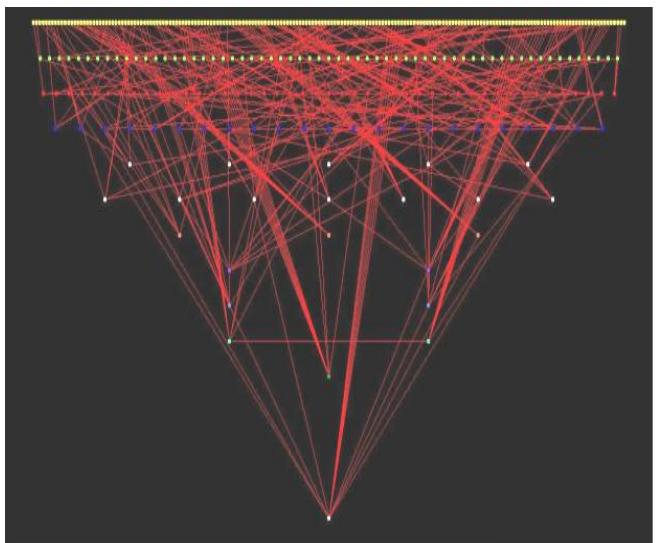


Figure 4 Hierarchical representation of nodes based on degree

TABLE IV. BEFORE AND AFTER GENERATING NEW EDGES

Measure	Before	After	% change
Closeness	0.0741	0.11	48.4
Average Talk time	152.3 units	198.6 units	30.4
Average Revenue	228.45	300.7	31.6
Betweenness	0.443	0.441	(0.45)
Modularity	0.992	0.9910	0.08

Fig. 5 shows the triads formed on a community of size 339 nodes with weight threshold 100. Red colored edges (353) are the existing set of edges and yellow colored edges (311) are new edges on addition of which triads are formed.

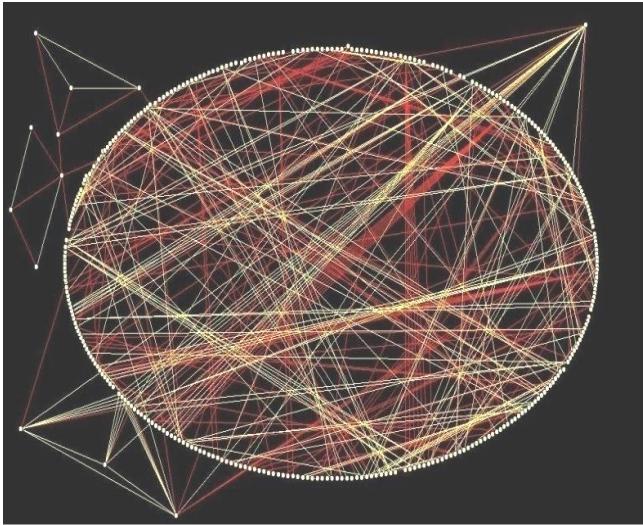


Figure 5 Triads formed on a community

V. DISCUSSION

In this section, we discussed about the advantages of the research based on our experiment of triad formation with a community of size 339 nodes as an example. Various graph theory attributes [2] which depict the strength of a network have been calculated and compared under two scenarios namely before generation of new edges and after addition of new edges.

The attributes which were used to show the success of our method are *Closeness*, *Average talk time of the community*, *Average Revenue of the community*, *Betweenness* and *Modularity*. Closeness is the inverse of sum of the shortest distances between each and every node in the network. Betweenness is a measure which depicts how much the network can be shattered if a person is inactive or churned.

From Table IV, we infer the following:

- 1) Closeness between nodes is increased by 48.4% from the earlier case which implies of how close the nodes are knitted. The shortest path between any two nodes has decreased and hence there is an increase in the measure
- 2) Average Talk time has increased by 30.4% as new edges lead to rise in number of calls.
- 3) Average Revenue has increased by 31.6%. Number of calls has increased which will automatically trigger the revenue to higher levels.
- 4) Betweenness has decreased by 0.45% which shows that the amount of shatter or disturbance caused by a particular user has lessened.
- 5) There is not much increase in modularity as our model was of size 339 nodes and modularity would only increase if there are new users added to the community. In our cases we are only working on new edges. The increase of 0.08% can be explained as formation of one or two isolated communities as new edges have been formed.

VI. CONCLUSION

Based on the results we can conclude that all the steps followed in our experiment have enriched the results starting from the initial step of filtering out users who's either indegree or outdegree is 0. Then our assumption of concentrating on active users would lead to better results has also been proven by achieving communities of higher modularity. Generation of new edges have been proved to be an advantage for the operator as it bonds its users much higher than the earlier stage which would in turn increase his revenue. Considering other parameters of CDR like SMS, GPRS, would lead us to find much more potential nodes which later can be used as data for our experiments. Moreover the Customer ranking can be considered at community level. The usage of location-based information can strengthen our claim of triad formation in mobile networks.

REFERENCES

- [1] S. Doyle, "Social network analysis in the Telcom sector Marketing applications" Journal of Database Marketing & Customer Strategy Management (2008) 15, 130 – 134. doi: 10.1057/dbm.2008.8
- [2] M. Saravanan, G. Prasad, S. Karishma and D. Suganthi, "Labeling communities using structural properties," International Conference on Advances in Social Networks Analysis and Mining, 2010. doi: 10.1109/ASONAM.2010.49
- [3] K. Wakita and T. Tsurumi . "Finding community structure in a mega-scale social networking service". in Proceedings of IADIS international conference on WWW/Internet 2007, pp.153-162, 2007.
- [4] A. Clauset, M. E .J. Newman, and C. Moore, "Finding community structure in very large networks". Phys. Rev. E 70, 066111, 2004. doi: 10.1103/PhysRevE.70.066111
- [5] M. Granovetter, "The strength of weak ties," The American Journal of Sociology, vol. 78(6), pp. 1360–1380, 1973.
- [6] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," Phys. Rev. E 69,026113, 2004. doi: 10.1103/PhysRevE.69.026113
- [7] S. Milgram, "The small world problem," Psychology Today, 1967.
- [8] A. Bookstein, "Informetric distributions, part i: Unified overview," American Society for Information Science, Vol. 41, pp. 368–375, 1999. doi: 10.1002/(SICI)1097-4571(199007)
- [9] R. L. Vincent D. Blondel, J. L. Guillaume and E. Lefebvre, "Fast unfolding of communities in large networks," Vol. 2008, No.10. doi: 10.1088/1742-5468/2008/10/P10008
- [10] M. E. J. Newman "Modularity and community structure in networks", physics/0602124 Proceedings of the National Academy of Sciences (USA) 103, pp. 8577—8582, 2006.
- [11] <http://vlado.fmf.uni-lj.si/pub/networks/pajek/>.

Ambients of Persistent Concurrent Objects

Suad Alagić

*Department of Computer Science
University of Southern Maine
Portland, Maine, USA
alagic@usm.maine.edu*

Akinori Yonezawa

*Department of Computer Science
University of Tokyo
Tokyo, Japan
yonezawa@is.s.u-tokyo.ac.jp*

Abstract—This paper develops a typed object-oriented paradigm equipped with message-based orthogonal persistence. Messages in this paradigm are viewed as typed objects. This view leads to a hierarchy of types of messages that belong to the core of typed reflective capabilities. Unlike most persistent object-oriented models, this model is equipped with general integrity constraints that also appear as a hierarchy of types in the reflective core. A transaction is naturally viewed as a sequence of messages and it is equipped with a precondition and a postcondition. The presented framework is motivated by ambients of persistent concurrent and mobile objects. The most important practical results supporting the developed model are verification techniques and a virtual platform for constraint management.

Keywords-Object databases; constraints; reflection; transactions.

I. INTRODUCTION

The current object technology has nontrivial problems in specifying classical database integrity constraints, such as keys and referential integrity [10][13][14]. No industrial database technology allows object-oriented schemas equipped with general integrity constraints. In addition to keys and referential integrity, such constraints include ranges of values or number of occurrences, ordering, and the integrity requirements for complex objects obtained by aggregation [1]. More general constraints that are not necessarily classical database constraints come from complex application environments and they are often critical for correct functioning of those applications [2].

Object-oriented schemas are generally missing database integrity constraints because those are not expressible in type systems of mainstream object-oriented programming languages. Since the integrity constraints cannot be specified in a declarative fashion, the only option is to enforce them procedurally with nontrivial implications on efficiency and reliability. The constraints must fit into type systems of object-oriented languages and they should be integrated with reflective capabilities of those languages [15]. Most importantly, all of the above is not sufficient if there is no technology to enforce the constraints, preferably statically, so that expensive recovery procedure will not be required when a transaction violates the constraints at run-time [1][2].

The object-oriented database model presented in this paper integrates message-based orthogonal persistence, object-oriented schemas equipped with general integrity constraints accessible by reflection, and transactions that are required to satisfy the schema integrity constraints. The model is based on a type system and it offers a significantly different view of messages in comparison with the mainstream object-oriented languages. The model applies to ambients of persistent and concurrent objects.

A message in mainstream object-oriented languages such as Java or C# is specified in a functional notation. This functional view fits messages that cause no side-effects and report the properties of the hidden object state. The functional view also fits queries. Other categories of messages do not fit the functional notation. An update message is a message that changes the state of the receiver and possibly other objects as well. An update message does not have a result and its semantics does not fit the functional notation.

An asynchronous message [18] in general does not have a result either and hence the functional notation is not appropriate. A particular type of an asynchronous message (a two-way message) has a result, but this result is not necessarily immediately available at the point of the message send. Asynchronous (remote) queries would fit this pattern. A transient message has a limited lifetime and a sustained message does not have this limitation. A message may be one-to-one with a single receiver or a message may be a broadcast message sent to a set of receiver objects. Many messages naturally combine the features of the above mentioned message types. For example, a two-way transient message, a one-to-one query message, a one-to-many sustained update message etc. [9].

Further development of this approach leads to an orthogonal model of persistence [5] that is based on a special message type that promotes the receiver object to persistence. A transaction is defined as a sequence of messages of different types. Concurrency control and recovery protocols can now be implemented in the object-oriented style. Indeed, serialization protocols require knowledge of types of messages (queries versus updates) and impose an appropriate ordering of conflicting messages. Similar comments apply to recovery protocols that are in our view sequences of do, undo and redo

messages.

Object-oriented constraints are a key feature of the presented model. Specifying the behavior of objects of a message type is naturally done using an object-oriented assertion language. Object-oriented assertion languages allow specification of database integrity constraints as class invariants, declarative specification of transactions with pre and post conditions, and queries whose filtering (qualification expression) is specified as an assertion predicate. The assertion languages used to express constraint-related features of the model presented in this paper are JML (Java Modeling Language) [11] and Spec# [12].

Two critical pieces of the technology that supports this model are extended virtual platform for constraint management and verification techniques that apply to constraints. The extended virtual machine integrates constraints into the run-time type system, allows their introspection and enforcement [15]. Verification techniques apply to object-oriented transactions written in Java or C#. The verification technologies are based on PVS [2] and automatic static techniques of Spec# [1].

We first present in Section II a motivating application based on ambients of concurrent and service objects. The fundamentals of the view of messages as typed objects is developed further in Section III, along with the hierarchy of message types. The model of persistence is described in Section IV. Queries and transactions are discussed in Section V. Type safe reflection which includes run-time representation of types (including message types) and assertions is the subject of Section VI.

II. MOTIVATING APPLICATION: AMBIENTS OF CONCURRENT OBJECTS

In this introductory section, we describe the environments that lead to the view of messages as typed objects. An ambient [9] is a dynamic collection of service objects. The types of service objects are assumed to be derived from the type *ServiceObject*. This is why the class *Ambient* is parametric and its type parameter has *ServiceObject* as its bound type as follows:

```
abstract class Ambient
<T extends ServiceObject> { . . . }
```

When a message is sent to an ambient object, one or more service objects is selected depending upon the type of the message, and the message is sent to those service objects. Messages sent to an ambient are in general asynchronous, hence they are of the type *Message*. When such a message object is created, it has its identity, a lifetime, and behaves according to one of the specific subtypes of the type *Message*. For example, a transient message has a limited discovery time and a sustained message does not. Moreover, messages can be sent to message objects. For example, if a message is a two-way message, a message that refers to the

future method may be sent to the two-way message object to obtain the result when it becomes available [18].

An ambient has a filter which selects the relevant service objects that belong to the ambient. This predicate is defined for a specific *Ambient* class, i.e., a class that is obtained from the class *Ambient* by instantiating it with a specific type of service objects. An ambient has a communication range which determines a collection of service objects that are in the ambient's range. The reach of an ambient object is then the collection of all service objects of the given type that satisfy the filter predicate and are within the communication range of the ambient object.

The class *Ambient* is equipped with a scheduler which selects the next message for execution according to some strategy. So the *Ambient* class looks like this:

```
abstract class Ambient
<T extends ServiceObject> {
    abstract boolean filter(T x);
    Set<Message> messages();
    Set<T> communicationRange();
    Set<T> reach();
    invariant (forAll T x)
        (x in this.reach() <=>
            this.filter(x) and x in
            this.communicationRange());
}
```

An example of a specific ambient class is

```
class StockBroker extends ServiceObject {
    int quote(String stock);
    int responseTime();
    . . .
}
class StockBrokerAmbient
    extends Ambient<StockBroker> {
    String[] displayStocks(){. . .};
    requestQuote(String stock){. . .};
    boolean filter(StockBroker x)
        {return x.responseTime() <=10;};
}
StockBrokerAmbient stockbrokers =
    new StockBrokerAmbient();
```

In a more general concurrent setting [18], a concurrent object is equipped with its own virtual machine. A virtual machine is equipped with a stack, a heap, a queue of messages, and a program counter (PC) as shown in Figure 1.

```
interface ConcurrentObject { . . . }
class ConcurrentObjectClass
    implements ConcurrentObject {
    private VirtualMachine VM();
}
```

In a concurrent paradigm of [18], a concurrent object executes messages that it receives by invoking the corresponding methods. In order to be able to do that, the heap of the object's virtual machine must contain reflective classes such as *Class*, *Method*, *Message* etc. These classes are

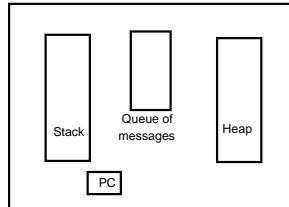


Figure 1. A concurrent object

stored on the heap of the object's virtual machine. The heap also holds the object state. Execution of a method is based on the object's stack according to the standard stack-oriented evaluation model.

A concurrent object gets activated by receiving a message. If a concurrent object is busy executing a method, the incoming message is queued in the message queue of the object's virtual machine. Messages in the queue will be subsequently picked for execution when an object is not busy executing a method. So at any point in time an object is either executing a single message or else it is inactive (i.e., its queue of messages is empty).

In the extreme case, all objects are concurrent objects, i.e., the class `ConcurrentObjectClass` is identified with the class `Object`. A service object is now defined as a concurrent object:

```
interface ServiceObject
    extends ConcurrentObject { . . . }
```

We can now redefine an ambient in this new setting as a concurrent object which represents a dynamic collection of concurrent service objects:

```
class Ambient <T extends ServiceObject>
    extends ConcurrentObject {
    . . .
}
```

Since an ambient is a concurrent object, it has its own virtual machine with a queue of messages sent to the ambient object and not serviced yet.

A mobile object is a concurrent object that is equipped with a location:

```
interface MobileObject
    extends ConcurrentObject{
    Location loc();
}
```

A region is an ambient that captures the notion of locality. It consists of all concurrent objects within the region as well as the service objects in that region, as illustrated in Figure 2.

```
class Region <T> extends Ambient<T> {
    Set<ConcurrentObject> objects();
    boolean withinRegion(MobileObject x);
    invariant (ForAll MobileObject x)
        (this.withinRegion(x) =>
```

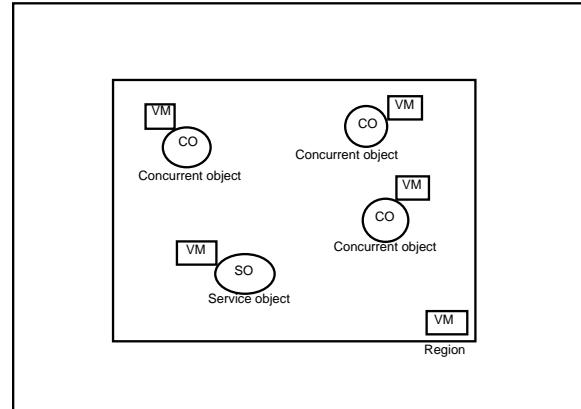


Figure 2. Regions of concurrent and service objects

```
    x in this.objects());
}
```

For example, if `class Server extends ServiceObject { . . . }` then `Region<Server>` would be an example of a region type. Since a region is a concurrent object, it is equipped with its own virtual machine. Also, since a region is an ambient, it receives messages that are queued in the message queue of the region's virtual machine to be serviced. Servicing a message sent to a region amounts to selecting a server object and sending the message to that server.

III. TYPES OF MESSAGES

Non-functional messages in this paradigm are objects. A message is created dynamically and it has a unique identifier like any other object. In the concurrent architecture described in Section II object identifiers must be global. The attributes of a message are the receiver object and the array of arguments along with a reference to a method. Messages of specific subtypes will have other attributes. This produces a hierarchy of message types that are subtypes of the type `Message`:

```
interface Message {
    Method m();
    Object receiver();
    Object[] arguments();
    int timeStamp();
}
```

When a message object is created its time stamp is recorded. The implementing class would have a constructor:

```
class MessageObject implements Message {
    MessageObject(Method m, Object receiver,
                 Object[] arguments);
    int timeStamp();
    Method m();
    Object receiver();
    Object[] arguments(); }
```

Copyright (c) IARIA, 2011

ISBN:978-1-61208-002-4

157

Creating a message could be done just like for all other objects:

```
Message msg =
new MessageObject(Method m, Object receiver,
Object[] arguments)
```

This implies message send in the underlying implementation. However, `Message` and `MessageObject` belong to the reflective core along with `Class`, `Method`, and `Constructor`. These types should be final in order to guarantee type safety at run-time. So an alternative is to have a special notation to create an asynchronous message. A functional (and hence synchronous) message is denoted using the usual dot notation:

```
x.m(a1,a2,...,an).
```

A non-functional (asynchronous etc.) message would be created as follows:

```
Message msg = x<=m(a1,a2,...,an).
```

In general, an asynchronous message does not have a result. The basic type of a message is point-to-point, one-way, and immediately executed. This type of a message could be expressed in a traditional notation

```
receiver.m(arguments)
```

In the new paradigm, the result of an asynchronous message send is a reference to the created message object. An example is:

```
Method requestQuote =
getClass(''StockBrokerAmbient'').getMethod(
    'requestQuote', getClass(''String''));
Message requestQuoteMsg =
    new MessageObject(requestQuote,
        stockBrokers, stock);
```

An alternative notation looks like this:

```
Message requestQuoteMsg =
    stockbrokers <= requestQuote(stock);
```

An update message is a message that mutates the state of the receiver object and possibly other objects as well. An update message does not have a result, hence we have:

```
interface UpdateMessage extends Message {...}
```

A special notation for an update message is

```
x<:=m(a1,a2,...,an)
```

The type of this expression is `UpdateMessage`.

A two-way message requires a response which communicates the result of a message. The result is produced by invoking the method `future` on a two-way message [18]. This method has a precondition which is that the future is resolved, i.e., that it contains the response to the message.

```
interface TwoWayMessage extends Message{...}
```

The implementing class would contain a constructor which takes the reply interval as one of its parameters.

```
class TwoWayMessageObject
```

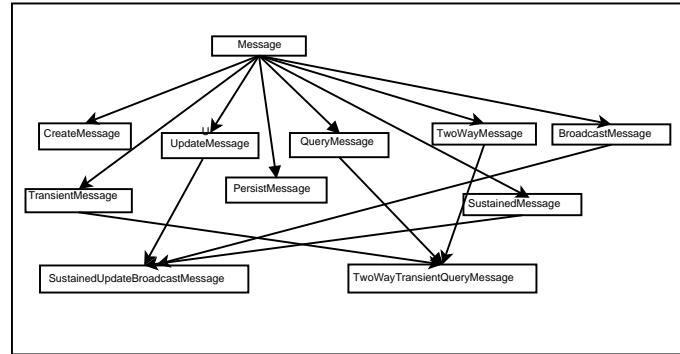


Figure 3. Message type hierarchy

```
implements TwoWayMessage {
    TwoWayMessageObject(Method m,
        Object receiver, Object[] arguments,
        int replyInterval);
    boolean futureResolved();
    boolean setFuture();
    Object future()
        requires this.futureResolved();
}
```

An example of a two way message is:

```
TwoWayMessage requestQuoteMsg =
    new TwoWayMessageObject(requestQuote,
        stockBrokers, stock, 20);
```

A suggestive notation for a two way message is:

```
TwoWayMessage requestQuoteMsg =
    stockbrokers <=> requestQuote(stock, 20);
```

A one-to-many message is of the type `BroadcastMessage` and it is sent to multiple objects. Using a suggestive notation for a one-to-many message, we would have:

```
Message requestQuoteMsg =
    stockbrokers <<=> requestQuote(stock);
```

A transient message has a discovery time specified as a finite time interval. If a message is not discovered and scheduled for execution before its discovery time has expired, the message will be regarded as expired and will never be scheduled for execution. The discovery time will be specified in the constructor of the implementing class. A suggestive notation for a transient message is `<=|`. A sustained message (i.e. a message whose discovery time is not limited) denoted as `<=~` is specified by a special message type `SustainedMessage`.

IV. PERSISTENT OBJECTS

An object is promoted to persistence by executing a message `persist` which specifies a user name and a name space. This message binds the object to the given user name in the given name space. The root class `Object` is equipped

with a method `persist` which means that the model of persistence is orthogonal, i.e., objects of any type may be promoted to persistence.

```
class Object { . . .
void persist(NameSpace scope, String userID); }
```

A name space consists of bindings of user names to objects. Name spaces can be nested. A name space is equipped with methods for establishing such a binding and for looking up an object in a name space bound to a given user id. Typically, name spaces are persistent.

```
interface NameSpace extends ConcurrentObject{
boolean bind(Object x, String name);
Object lookup(String name);
}
```

The type `PersistMessage` is now defined as follows:

```
interface PersistMessage extends Message {
NameSpace scope();
Object userID(String name);
}
```

Creation of a `persist` message is denoted by a special notation using the symbols `<=?persist`.

A schema extends a name space with additional methods. One of them is the method `select` that returns a set of objects in the schema that satisfy a given assertion.

```
interface Schema extends NameSpace { . . .
Set<Object> select(Assertion a);
}
```

The integrity constraints of a schema are specified in its invariant as illustrated in the example below. The schema `StockMarket` is equipped with a key constraint and a referential integrity constraint.

```
interface Stock {
String code();
float price();
}
interface Broker {
String name();
Set<Stock> stocks();
}
interface StockMarket extends Schema {
Set<Stock> stocks();
Set<Broker> brokers();
invariant:
(forAll s1,s2 in this.stocks():
s1.code() == s2.code() => s1.equals(s2));
(forAll b in this.brokers():
(forAll sb in b.stocks():
(exists s in stocks():
(sb.code() == s.code()))));
}
```

As for a specific assertion language, our previous results such as [2][4] are based on JML and more recent experiments are based on Spec# [1][6]. In fact, our extended

virtual platform [15] accommodates a variety of assertion languages.

V. QUERIES AND TRANSACTIONS

A query message is specified below as an asynchronous message. Its type is a subtype of `TwoWayMessage`. So the result of a query may not be immediately available. When it is, it will be available by sending a functional message future to the query message object.

```
interface QueryMessage
extends TwoWayMessage {
Schema scope();
Assertion query();
}
```

Creation of a particular query object is illustrated below using a special notation with the symbol `<=?select`:

```
StockMarket sch; QueryMessage q;
q <=?select(
    forAll b in sch.brokers():
        (exists s in b.stocks():
            s.code() == ``SNP500''));
```

A database server is a specific subtype of a service (and hence concurrent) object. It implements a schema:

```
interface DbServer
extends ServiceObject, Schema {
Sequence<Message> log();
}
```

Since a database server is a concurrent object, it is equipped with its own virtual machine. Typically, a database server is a persistent concurrent object. Hence by reachability, its schema (which includes persistent objects and integrity constraints) and its virtual machine will also be persistent.

A database server is equipped with a log of received messages. Here the view of messages as typed objects is critical. Committing a transaction requires extraction of the update and persist messages to reflect those changes in database collections. Implementing serializability protocols requires distinguishing update and query messages and controlling the order of their execution. All of this is possible because these messages are objects belonging to different types so that their properties can be inspected by sending functional messages to those objects.

Unlike the ODMG model, a transaction type is parametric. Its bound type specifies that the actual type parameter must be derived from the interface `Schema`.

```
interface Transaction<T extends Schema> {
boolean commit();
boolean abort();
}
```

Another distinctive feature of the notion of a transaction with respect to ODMG and other persistent object models is that a transaction is naturally equipped with a precondition

and a postcondition and it is defined as a sequence of messages of different types (such as query, update and persist messages). An illustrative example is:

```
interface StockUpdate
    extends Transaction<StockMarket> {
    StockMarket schema();
    void update(String stockCode, float value)
    requires (exists s in this.schema().stocks():
              s.code() == stockCode);
    ensures (forAll s in this.schema().stocks():
              s.code() == stockCode =>
                      s.price() == value);
}
```

The implementing class of the interface Transaction would have the following form:

```
class TransactionObject<T>
    implements Transaction<T extends Schema>{
    TransactionObject(T dbSchema);
    Sequence<Message> body();
    boolean commit();
    boolean abort();
}
```

Taking this approach one step further, a transaction is a concurrent object defined as follows:

```
class ConcurrentTransactionObject<T>
    implements ConcurrentObject
    implements Transaction<T extends Schema>
{ . . . }
```

VI. REFLECTION

Just like in Java Core Reflection (JCR), reflection in a language that supports messages as typed objects includes classes Class, Method, and Constructor. The main differences in comparison with JCR are:

- Reflection includes the interface Message with its various subtypes.
- Reflection includes the interfaces Assertion and Expression with their various subtypes.

The core reflective class Class has the following abbreviated signature. A distinctive feature is an assertion representing a class invariant.

```
class Class { . . .
    String name();
    Method[] methods();
    Method getMethod(String name,
                     Class[] arguments);
    Assertion invariant();
}
```

The reflective class Method is defined as follows. Its distinctive features are a pre condition and a post condition expressed as assertions. Their type is Assertion.

```
Class Method { . . .
    String name();
    Class declaringClass();
    Assertion preCondition();
```

```
Assertion postCondition();
Class[] arguments();
Class result();
Expression body();
Object eval(Object receiver,
            Object[] args);
}
```

The body of a method is an expression evaluated by the function eval. Just like Assertion, the type Expression belongs to the reflective core. The method eval evaluates the method body after binding of variables occurring in the expression representing the method body is performed. The variables to be supplied to eval are the receiver and the arguments.

Availability of assertions in the classes Method and Class is a major distinction with respect to the current virtual machines such as JVM or CLR. This is at the same time a major difference with respect to the assertion languages such as JML or Spec#. Full implementation of this distinction is given in our previous work [15].

VII. RELATED RESEARCH

The orthogonal model of persistence implemented in [5] and the ODMG model of persistence [8] are based on promoting an object to persistence by either binding it to a name in a persistent name space or making it a component of an object that is already persistent. Message-based model of persistence presented in this paper is a further significantly different development after these initial approaches.

In the ODMG model queries and transactions are objects, and so are in our model, with additional subtleties. In our approach messages are objects, and queries and updates are particular types of messages. A transaction is a concurrent object which consists of a sequence of messages. The fact that messages are objects makes it possible to construct a transaction log as a sequence of messages of different types (queries and updates, checkpoints, commits etc.).

General integrity constraints are missing from most persistent and database object models with rare exceptions such as [1][4][7]. This specifically applies to the ODMG model, PJama, Java Data Objects, and just as well to the current generation of systems such as Db4 Objects [10], Objectivity [14] or LINQ [13]. Of course, a major reason is that mainstream object-oriented languages are not equipped with constraints. Those capabilities are only under development for Java and C# [6][11].

Constraints in the form of object-oriented assertions are a key component of our approach. Database integrity constraints are specified as class invariants, transactions are specified via pre and post conditions, and queries come with general filtering (qualification) predicates. In comparison with object-oriented assertion languages, such as JML [11] and Spec# [6][12], a major difference is that in our approach assertions are integrated in the run-time type system and

visible by reflection. This makes database integrity constraints accessible and enforceable at run-time. Reflective constraint management, static and dynamic techniques for enforcing constraints, and transaction verification technology are presented in [2][4][15].

Our sources of motivation for the view of concurrent, distributed and mobile objects were the languages ABCL [17][18] and AmbientTalk [9]. The core difference is that both of the above languages are untyped, whereas our approach here is based on a type system. A further distinction is that ABCL and AmbientTalk are object-based and our approach is class based. Other related work is given in [16]. Unlike ABCL reflective capabilities, reflection in this paper is type-safe. A major distinction is the assertion language as a core feature of the approach presented in this paper.

Verification techniques of object-oriented transactions with schemas and transactions specified in either JML or Spec# are presented in [1][2].

VIII. CONCLUSION

Object-oriented assertions allow specification of object-oriented schemas equipped with database integrity constraints, transactions and their consistency requirements, and queries. The view of messages as typed objects leads to a typed reflective paradigm equipped with a message-based orthogonal persistence.

Integrating the above features into the reflective capabilities of the virtual platform leads to static and dynamic techniques for enforcing database integrity constraints. Reflection in this paradigm is much more general than reflection in main-stream typed object-oriented languages as it includes message and assertion types that are integrated into the run-time type system.

The presented approach requires more sophisticated users that can handle object-oriented assertion languages such as JML or Spec#. Those languages and their underlying technologies come with nontrivial subtleties as they are still in the prototype phase. Integrating these technologies into existing object database systems presents a significant challenge yet to be addressed in our future research.

One the other hand, the benefits of the availability of general constraints and static verification of transactions with respect to those constraints are very significant. Data integrity as specified by the constraints could be guaranteed, runtime efficiency and reliability of transactions is significantly improved, and expensive recovery procedures will not be required for constraints that were statically verified. In addition, more general application constraints that are not necessarily database constraints could be guaranteed. All of this produces a much more sophisticated technology in comparison with the existing ones.

REFERENCES

- [1] S. Alagić, P. Bernstein, and R. Jairath, Object-oriented constraints for XML Schema, Proceedings of ICOODB 2010, *Lecture Notes in Computer Science 6348*, pp. 101-118.
- [2] S. Alagić, M. Royer, and D. Briggs, Verification technology for object-oriented/XML transactions, Proceedings of ICOODB 2009, *Lecture Notes in Computer Science 5936*, pp. 23-40.
- [3] S. Alagić, The ODMG object model: does it make sense?, Proceedings of OOPSLA, pp. 253-270, ACM, 1997.
- [4] S. Alagić and J. Logan, Consistency of Java transactions, Proceedings of DBPL 2003, *Lecture Notes in Computer Science 2921*, pp. 71-89, Springer, 2004.
- [5] M. Atkinson, L. Daynes, M. J. Jordan, T. Printezis, and S. Spence, An orthogonally persistent JavaTM, ACM SIGMOD Record 25, pp. 68-75, ACM, 1996.
- [6] M. Barnett, K. R. M. Leino, and W. Schulte, The Spec# programming system: an overview, Microsoft Research 2004, <http://research.microsoft.com/en-us/projects/specsharp/> [retrieved, November 16, 2010].
- [7] V. Benzaken and X. Schaefer, Static integrity constraint management in object-oriented database programming languages via predicate transformers, Proceedings of ECOOP '97, *Lecture Notes in Computer Science 1241*, pp. 60-84, 1997.
- [8] R. G. G. Cattell, D. Barry, M. Berler, J. Eastman, D. Jordan, C. Russell, O. Schadow, T. Stanienda, and F. Velez, *The Object Data Standard: ODMG 3.0*, Morgan Kaufmann, 2000.
- [9] T. Van Cutsem, Ambient references: object designation in mobile ad hoc networks, Ph.D. dissertation, Vrije University Brussels, 2008.
- [10] Db4 objects, <http://www.db4o.com> [retrieved, November 16, 2010].
- [11] G. T. Leavens, E. Poll, C. Clifton, Y. Cheon, C. Ruby, D. Cook, P. Muller, and J. Kiniry, JML Reference Manual, <http://www.eecs.ucf.edu/~leavens/JML/> [retrieved, November 16, 2010].
- [12] K. R. Leino and P. Muller, Using Spec# language, methodology, and tools to write bug-free programs, Microsoft Research, <http://research.microsoft.com/en-us/projects/specsharp/> [retrieved, November 16, 2010].
- [13] LINQ: Language Integrated Query, <http://msdn.microsoft.com/en-us/library/bb308959.aspx> [retrieved, November 16, 2010].
- [14] Objectivity, <http://www.objectivity.com/> [retrieved, November 16, 2010].
- [15] M. Royer, S. Alagić, and D. Dillon, Reflective constraint management for languages on virtual platforms, *Journal of Object Technology*, vol 6, pp. 59-79, 2007.
- [16] J. Schafer and A. Poetzsch-Heffter, JCoBox: Generalizing active objects to concurrent components, Proceedings of ECOOP 2010, *Lecture Notes in Computer Science 6183*, pp. 275-299.
- [17] T. Watanabe and A. Yonezawa, Reflection in an object-oriented concurrent language, Proceedings of OOPSLA, pp. 306-315, ACM Press 1988.
- [18] A. Yonezawa, J.-P. Briot, and E. Shibayama, Object-oriented concurrent programming in ABCL/1, Proceedings of OOPSLA, pp. 258-268, ACM Press 1986.

Modeling Temporal Databases and Temporal Constraints

Mohamed Mkaouar, Mohamed Moalla

LIP2 Laboratory

University of Tunis El Manar, Faculty of Science

Campus Universitaire 2092 - El Manar Tunis, Tunisia

Mkaouar.Mohamed@gmail.com, Mohamed.Moalla@fst.rnu.tn

Rafik Bouaziz

MIRACL Laboratory

University of Sfax, Faculty of Economics and Management

Route de l'Aéroport 3018, Sfax, Tunisia

Raf.Bouaziz@fsegs.rnu.tn

Abstract— Applications requiring a full and efficient management of data feel the need to consider, beside the current facts, historical and future facts, and to keep the track of the manipulation of facts by the DBMS. In this paper, we define concepts and modeling tools to express constraints, taking into account the temporal dimension. The expression of these constraints is achieved through an independent platform modeling, in the UML-TF profile. Next, we propose extensions to the SQL3 to be able to convert enhanced UML-TF representations in a specific object-relational platform.

Keywords-Modeling Temporal Databases; Temporal Constraints; UML profile; SQL3 extension

I. INTRODUCTION

Over the past twenty five years there have been many studies concerning the integration of different temporal specifications in databases (DB), and of new languages and temporal features in the DBMS [11][14][15][25]. Nevertheless, there is not yet a general implementation of Temporal DBMS and DB by manufacturers and designers. We attribute this fact to two main reasons: the importance of legacy DB and the complexity of the temporal environment.

Indeed, the importance of legacy DB, and of applications that exploit them, makes any translation from a modeling and/or development environment to another difficult and expensive. That explains the slow transition from navigational (network) DB to relational DB, and why, despite the dominance of object-oriented programming, OODB have yet to find dominance. It is the same for the temporal dimension; there is now a gentle introduction to some temporal features in the current DBMS [16][20][21], which are still limited.

Investigations concerning works dealing with temporal environment are not yet sufficient to develop temporal DBMS. Other investigations remain indispensable to master temporal data management and schema evolution [2], concurrency control in temporal DB [17], development of temporal applications [18], etc.

Such development requires appropriate formalisms and tools, as well as a solid training for developers to this new environment. It also requires rigorous use of the principles of abstraction, provided by systemic methods and adopted by object-oriented approaches, especially with the advent of the MDA (Model Driven Architecture) introducing three levels of abstraction: CIM (Computational Independent Model),

PIM (Platform Independent Model) and PSM (Platform Specific Model).

We must therefore define a CIM and a PIM before proceeding within a development considering a specific platform in order to simplify the adoption of temporal DB. But this principle of abstraction has not been adopted by most of the works on temporal DB.

In this paper, we start by enriching the UML-TF profile [19] to be able to model different classes of constraints related to time, in the CIM and PIM levels. We then study how to transform an UML-TF representation into a specific object-relational representation, including temporal constraints.

The remainder of this paper is organized into six sections as follows. Section two provides a brief overview of the state of the art. The third Section describes UML-TF and reviews on how to model specifications incorporating the temporal dimension in this profile, with an illustration based on a real application. In the fourth Section, we present different classes of constraints on these specifications and how to express them using examples from the same application. In the fifth Section we propose a transformation of a UML-TF representation into a specific object-relational representation.

II. STATE OF THE ART AND RELATED WORK

Time can be taken into account in accordance with what is happening in the real world and/or in DB, whose updates can be done with some phase shifts. So, two standards time types, *valid time* and *transaction time* [13], and several kinds of temporal facts which are mainly, *Valid-time facts*, *Transaction-time facts* and *bitemporal facts*, are defined.

Valid-time facts may relate to the past, to the present or to the future. Each such fact is represented by a timestamp with their valid-times in reality. Thus, it becomes possible to maintain the history of all valid facts, which can be updated in real time with retroactive effect or postactive effect, but not to keep track of deletions and corrections of errors.

Transaction-time facts can keep track of the manipulation of facts by the DBMS, which timestamps them by the execution time of the transaction that manipulates each fact. This track covers insert operations, update operations — whether evolution updates or error correction — and delete operations. Transaction-time facts timestamps are defined according to the schedule adopted by the operating system and a granularity generally equal to the second, but could be thinner if necessary. Thus, it becomes possible to maintain

the history of all the facts, valid or erroneous, past or current, but not future. But only the current facts may be updated; updates can not be made here either with retroactive effect, or with postactive effect.

These valid or transaction histories have then insufficiencies. A complete history can be assured only if facts are timestamped by both valid and transaction-times, thus we obtain bitemporal facts. It then becomes possible to update the facts with retroactive or postactive effects, keep track of valid facts and erroneous ones, and distinguish between valid facts and erroneous ones. However, the management of these facts becomes increasingly complicated [2]. Then, the use of temporal dimensions must be justified by the needs of users. A temporal DB must contain conventional facts (non-temporal), when we do not need historical, valid-time facts, when we need a valid history of facts in reality, transaction-time facts, when we need a history that keeps track of the manipulation of facts by the DBMS, and bitemporal facts, when we want to have a complete history. In addition, the DBMS must include effective techniques to handle these kinds of facts [2][17][23], on the one hand, and design methods must fully adopt the principle of abstraction and include means for expressing temporal specifications [19], on the other hand.

We want to expand in this paper the enhancement of the UML-TF to be able to model temporal constraints in the CIM and PIM levels. We then propose to classify the constraints into several classes and sub-classes (cf. Section 4) and suggests ways of expression for each of these classes and subclasses. To our knowledge, this aspect has not been sufficiently addressed, either by the temporal DB models, or by the works dealing with constraint classification and checking [3][7][9].

To the works concerning PSM, we note here that the expression of temporal constraints under SQL [6][24]. Snodgrass [24] were limited to examining primary keys and foreign keys, without detailing the other constraints that we may declare at the CIM and PIM. Authors of reference [6] proposed to represent all the constraints, even simple constraints of not null fields or of columns domain's, by means of assertions without using standard SQL statements, which do not promote their adoption. We want to enrich SQL3 to allow the expression of temporal constraints, affecting different levels of abstraction, in a declarative and the simplest possible manner.

III. MODELING TEMPORAL DB WITH UML-TF

We first review the main notation allowing the modeling of temporal specifications with UML-TF [19] and then illustrate such modeling with a case study.

A. Temporal notation

With UML-TF, modeling is made through the various levels of abstraction proposed by MDA, while expressing the temporal dimensions of any facts by stereotypes that we classify into three categories: valid-time stereotypes, transaction-time stereotypes and bitemporal stereotypes. To these three categories we associate the following icons:

- ‘⌚’ which symbolizes the real world clock, and is intended to the first category of stereotypes. This icon is used at the CIM level of the MDA approach.
- ‘⌚💻’ which represents the clock of the machine, and is intended to the second category of stereotypes. This icon is used at the PIM level of MDA.
- ‘⌚⌚’ which symbolizes both clocks, and is intended to bitemporal stereotypes. Such icon results from any two declarations, first by a valid-time icon, then by a transaction-time icon.

To realize the UML-TF profile, we defined an abstract stereotype for each of the three categories of stereotypes. Each abstract stereotype is a realization of the Evolutionary Stereotype [8] that allows to aggregate new semantic definitions. An abstract stereotype is then characterized by appropriate meta-properties [19]. Some of these meta-properties (Calendar, Type, Granule) concern timestamps and are initialized by default values. These values, which can be modified depending on the context, are to be exploited by the temporal features that the DBMS should ensure to allow the attribution of timestamps. Other meta-properties allow expressing constraints on timestamps or on temporal instances. These constraints, which we will detail in Section 4, are to be included in the schema of the DB and must be supported by the DBMS.

Each of the valid-time stereotypes, transaction-time stereotypes and bitemporal stereotypes represents both a realization of the concerned abstract stereotype, with appropriate values of its meta-properties, and an extension of the concerned meta-class of the UML metamodel. Due to space limitations, we limit ourselves here to mention these stereotypes and explain their effect. Further details concerning their definition can be found in [19].

1) *Valid-time stereotypes*: Any need of a history according to the valid-time dimension is expressed in the CIM level, by one of the following valid-time stereotypes:

- <<VTA>> (Valid-Time Attribute): associated with an attribute, this stereotype models the need to keep all the valid values that have taken or will take effect in reality. Each value is stamped with its valid-time.
- <<VTAs>> (Valid-Time Association): associated with an association, with or without associative-class, this stereotype models the need to preserve all valid links that have taken or will take effect in reality. If such a need is limited to one direction of the association, we use the stereotype <<VTR>> (Valid-Time Role).
- <<VTC>> (Valid-Time Class): This stereotype can associate valid timestamps to each object of the class.
- <<VTGS>> (Valid-Time Generalization-Specialization): This stereotype can associate valid timestamps to each object of the concerned subclasses.

2) *Transaction-time stereotypes*: Such stereotypes can be used, at the PIM level, for any element to model the need to keep track of their instances by the DBMS. We define

here, as in the previous case, a stereotype for each type of element: <<TTA>> (Transaction-Time Attribute), <<TTAs>> (Transaction-Time Association), <<TTR>> (Transaction-Time Role), <<TTC>> (Transaction-Time Class) and <<TTGS>> (Transaction-Time Generalization-Specialization).

3) *Bitemporal stereotypes*: Any element for which is associated first a valid-time stereotype and second a transaction-time stereotype is considered bitemporal; the two stereotypes are systematically replaced by a bitemporal stereotype which contains the meta-properties of the two others and represented by the bitemporal icon. We define the following bitemporal stereotypes: <<BTA>> (BiTemporal Attribute) <<BTAs>> (BiTemporal Association) <<BTR>> (BiTemporal Role) <<BTC>>

(BiTemporal Class) and <<BTGS>> (BiTemporal Generalization-Specialization).

B. Case study

As an example, let us consider the UML-TF class diagram of Figure 1. At this level, we briefly describe the application in general, and we further detail, in the following sections, how to declare constraints related to time. This diagram, concerns a higher education institution system. A person can be a student, a teacher or both. A teacher may be responsible at most than nine modules. Between seven and fifteen modules are taught for any given audience. Any training session concerns a teacher, a module, a group of an audience or an audience. It is described by a day, an hour and a classroom number. Each student should have, for each module that he studies, two or three marks.

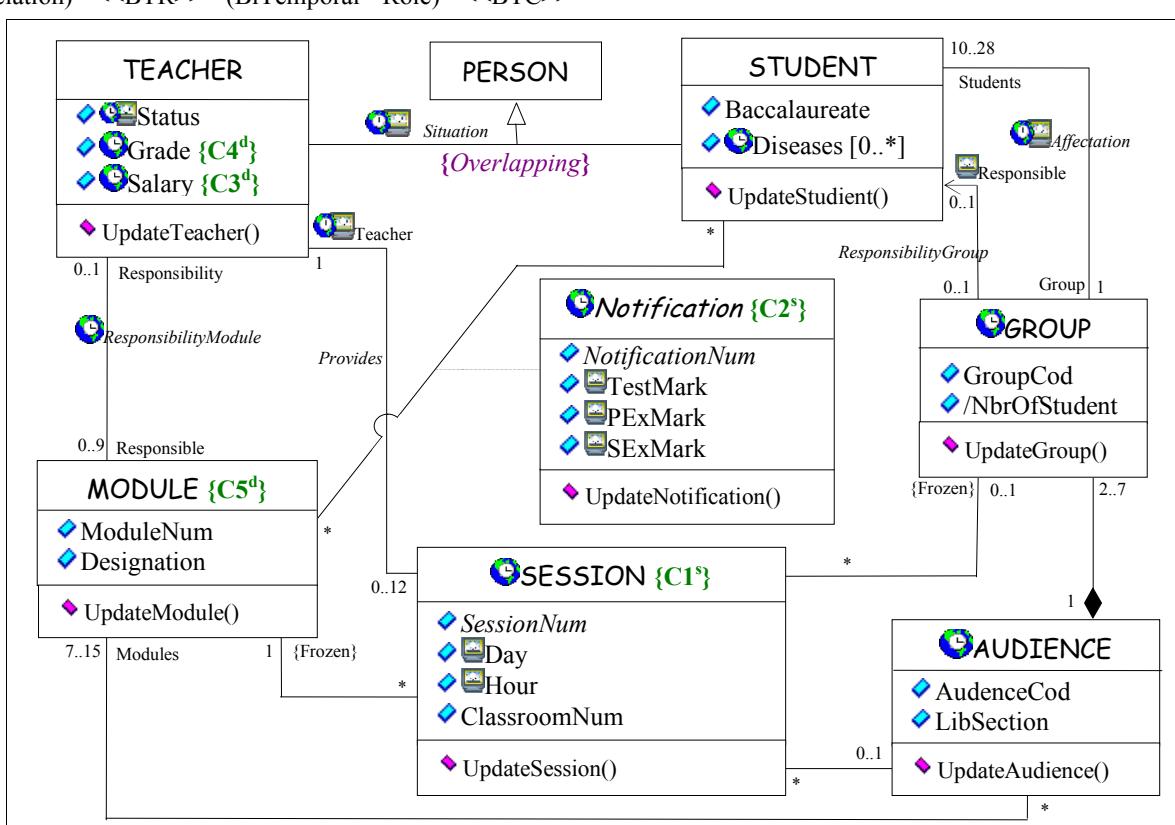


Figure 1. UML-TF class diagram modeling the DB of an application for a higher education institution.

In this diagram:

- ‘Grade’, ‘Salary’ and ‘Diseases’ are bitemporal attributes stereotyped by <<VTA>>;
- The association “ResponsibilityModule” and the associative-classes “Notification” are stereotyped by <<VTAs>>;
- AUDIENCE, GROUP and SESSION are three valid-time classes stereotyped by <<VTC>>;
- ‘Responsible’ is a transaction-time role stereotyped by <<TTR>>;
- ‘TestMark’, ‘PExMark’, ‘SExMark’, ‘Day’ and ‘Hour’ are transaction-time attributes stereotyped by <<TTA>>;
- ‘Status’ is a bitemporal attribute stereotyped by <<BTA>>;
- ‘Affectation’ is a bitemporal association;
- ‘Teacher’ is a bitemporal role stereotyped by <<BTR>>;
- TEACHER and STUDENT are bitemporal classes stereotyped by <<BTGS>>.

IV. NEW MEANS FOR EXPRESSING TEMPORAL CONSTRAINTS

Commonly, the constraints are classified into static and dynamic [10]. A static constraint controlling values or links that can take an attribute or a relationship, respectively, while a dynamic constraint control the evolution of these values or these links (the employee's salary can not decrease, for example).

For static constraints, we distinguish in UML simple constraints, called predefined, which usually focus on one element of the diagram and for which the notation defines "means" to express them in the diagram in a light manner. There are also complex static constraints, which are usually focused on more than one element of the diagram, and that we call here non-predefined constraints; using OCL [22], it is possible to formulate these constraints in different ways, depending on the context. To avoid overloading the diagrams, we propose to represent non-predefined constraints, and also dynamic constraints, not by their explicit expressions, but by codes assigned to them. These codes are to be placed between two brackets beside the name of the chosen context, as the five examples of the diagram in Figure 1 ($\{C1^s\}$, $\{C2^s\}$, $\{C3^d\}$, $\{C4^d\}$ and $\{C5^d\}$; it is possible to define many others temporal constraints in this diagram). To distinguish a static complex constraint from a dynamic constraint in the diagram, we use the exhibitors s , and d , next to the code of the constraint. The explicit expression in OCL of these constraints is then attached to the considered diagram.

To allow the expression of constraints on temporal instances, especially dynamic constraints, in OCL, it is required to extend this constraint language. Indeed, a temporal instance is characterized by elements from the following: Value, Start Valid-Time, End Valid-Time, Start Transaction-Time, End Transaction-Time and Index. The extension that we propose allows identifying these elements, by using the following keywords: **Value**, **SVT**, **EVT**, **STT**, **ETT** and **Index**. These keywords are to be used for any temporal instance using dot notation; the default is **Value**.

Following our analysis of the impact of the temporal dimension on the expression of constraints, we have proposed three categories of constraints related to time. We study these categories in the three following sub-sections.

A. Static constraints involving a temporal dimension

When a static constraint is declared on an element which is associated with a valid-time or a transaction-time stereotype, it often changes meaning; the DBMS is not sufficient to that verification at the current time, but extends this verification at any time of the maintained history to also cover past and, possibly, future facts.

The new meanings of these constraints do not affect the means of their expression, but rather affect their verification. However, we propose to express them by means of temporal invariants, denoted by '**Temporal inv:**'; we therefore propose to enrich our extension of OCL with this keyword.

Consider first the new meaning of an example of a predefined constraint of the diagram in Figure 1: "The

multiplicity of roles of the association "Affectation" means that at a given instant, a group of education is associated with at least 10 students and at most 28 students, and a student is assigned to one and a single group. However, a group may be associated with more than 28 students and a student may be assigned to multiple groups if we consider a history that stretches over a long period".

The constraint $\{\text{overlapping}\}$, which is not in fact, does not change meaning. Applied to the Generalization-Specialization *Situation*, it continues to mean that a person can be both a student and a teacher, either for the current time or at any past or future time.

We now detail the two non-predefined constraints $\{C1^s\}$ and $\{C2^s\}$.

- $\{C1^s\}$: A teacher can not teach the same group more than twice at the same period.

Context SESSION Temporal inv: C1

```
Session.allinstances->forAll(s1, s2, s3 | s1 <> s2
    and s2 <> s3 and s1.Audience = s2.Audience =
        s3.Audience implies s1.Teacher <> s2.Teacher
        or s2.Teacher <> s3.Teacher and s1.Group <>
            s2.Group or s2.Group <> s3.Group)
```

- $\{C2^s\}$: A mark is to attribute to a student for a given module only if this student is already enrolled in an audience for which the module is taught.

Context Notification Temporal inv: C2

```
(self.Module) includes
self.Student.Group.Audience.Modules
```

When these constraints are applied to bitemporal elements, only valid instances are taken into account. Incorrect or deleted (in a non-destructive manner) instances are not affected since they have already been verified.

B. Dynamic constraints

A formal expression of these constraints requires the use of a constraint language incorporating temporal operators, as already studied by several other works, especially those who have proposed temporal extensions to OCL [5][26]. This expression also requires the enrichment of OCL by the proposed keywords to be able to access to the various elements of a temporal instance. Nevertheless, for some constraints in this category, mainly those that focus on a single element, it is possible to find 'semi-graphic' means to their expression; this expression is easier to use in modeling.

Constraints $\{C3^d\}$, $\{C4^d\}$ and $\{C5^d\}$ are examples of dynamic constraints. In what follows, we detail their meaning and their expression in OCL extended by the above mentioned keywords and by operators of temporal logic. For the two constraints $\{C3^d\}$ and $\{C4^d\}$, we also propose 'semi-graphic' means to use as patterns for these types.

- $\{C3^d\}$: The salary of a teacher can not decrease.

Context TEACHER inv : C3

```
self.Salary->forAll(s1 : Salary, s2 : Salary |
    s1.SVT < s2.SVT implies s1.value < s2.value)
/*or s1 precedes s2 implies s1.value < s2.value) */
The 'semi-graphic' pattern that we propose to this type of constraint is the following: {↗}
```

- **{C4^d}** : The qualification of a teacher in his career follows the following order: ‘Assistant Professor’ (A), ‘Associate Professor’ (AP) and ‘Professor’ (Pr); but a teacher can begin as ‘AP’.

Context TEACHER inv : C4

```
self.Grade->forAll(g1 : grade, g2 : grade |
g1.SVT < g2.SVT implies g1.value = 'A' and g2
= 'AP' or g1 = 'AP' and g2 = 'Pr'
and self.Grad->forAll (g : grad | g.index =1
implies g = 'A' or g = 'AP')
```

The ‘semi-graphic’ pattern that we propose to this type of constraint is the following:

{Order : ‘A’, ‘AP’, ‘Pr’}

- **{C5^d}** : The responsibility of a module is assigned to a teacher only if this teacher involved in teaching this module since 5 years.

Context MODULE inv : C5

```
self.Responsability implies since 5 years
self.Responsability.Session->notEmpty()
```

C. Constraints on timestamps and on temporal instances

Compared to the two first categories of constraints, the constraints in this category may cover one or two temporal dimensions. These constraints concern stereotyped elements, as was announced in Section 2, and for which we have defined meta-properties in the abstract stereotypes. This allows expressing them in a simple manner, simply by valorizing the concerned meta-properties. We define two classes for this category: constraints on timestamps, and constraints on timestamped values of temporal instances.

1) *Constraints on timestamps*: In this class of constraints we distinguish three sub-classes as follows:

The constraints of the first sub-class concern the definition of default values or restrictions for a valid or transaction stamp. These constraints may relate to the timestamps of all instances of a stereotyped element, or for some instances of this element. Some restrictions depend on the properties of the timestamp, for example, it is not possible to restrict the duration of a timestamp of type instant.

As examples of such constraints, we can define for the considered application the following ones: “A teaching period starts on 15/9 of each year and ends on 31/7 of that year.”; “A teacher can not remain in the grade of ‘Associate Professor’ less than 3 years.”.

The second sub-class concerns all constraints defined between a valid timestamp and a transaction timestamp of a bitemporal element. These constraints have been widely studied in [12]. As examples, we define the following constraints: “The transaction-time of a status can not exceed its valid-time more than one month.”.

The constraints of the third sub-class are systematically imposed by the association of valid-time stereotypes in the diagram. These constraints depend on the type of the timestamp. In particular, it ensures data integrity when the timestamp type is ‘temporal interval’ or ‘temporal element’. Indeed, in this case:

- The timestamp of a link should be included (or equal) in (to) the timestamp of each concerned object; this is the case for example of links defined between STUDENT and GROUP.
- The timestamp of an object of an aggregate class must be included (or equal) in (to) the timestamp of the concerned object in the component class; this is the case of objects of the GROUP class vis-à-vis to objects of the AUDIENCE class.

- 2) *Constraints on temporal instances*: We retain at this level the following four main types:

- The extent of the maintenance of past valid values, in number of values or in duration relative to the current time; this concerns the *vacuuming* parameters [23] which is a way to destructively delete data becoming obsolete. The default values assigned to the two concerned meta-properties (V_Nbr and V_Duration) are equal to infinity. In the example, we can impose that: “The number of groups for each audience is not to historize beyond three periods.”; “The label of a section is to historize only for the last three values.”.
- The number of corrections or changes. By default, this number is unlimited. In the example, we can impose that it is not possible to correct the assignment of a training session to a teacher more than twice for the same period.
- How to keep the instances of a temporal collection vis-à-vis the *coalescing* operator [4]. By default, these instances are to keep coalesced, that is two instances that follow in time must have different values. In the example, links that concern affectations of students to groups, and notifications of students are to keep not coalesced.
- The management strategy for future data; this type of constraints is specific to bitemporal elements; by default, this management is to perform in a destructive manner.

V. SPECIFIC REPRESENTATION TO THE OBJECT-RELATIONAL PLATFORM

We present in this section how to extend the SQL3 standard by new keywords to allow the representation of timestamped tables and columns, as well as constraints related to time.

Our objective is to raise the issue and propose some solutions. More adequate solutions, especially as regards the expression of complex static constraints and dynamic ones, require substantial investment and resources that are beyond us.

A. Expression of timestamps

About the declaration of timestamps in SQL3 orders, we propose to put the letter ‘V’, the letter ‘T’ or the two letters, after the name of the concerned table or column. Each letter, representing a timestamp, is followed by the values of the meta-properties defined to the timestamp, when the defaults values have to be changed.

B. Static constraints involving temporal dimension

As in UML-TF, we propose that the expression of static constraints is to done in the same way, whether with or without a time dimension, apart from the fact that the syntax is enriched by the word '**Temporal**', to indicate that the constraint must be checked at any instant and not only at the current time. Indeed, the constraints checking module of the DBMS must ensure the checks within temporal DB.

To simplify the expression of different non-predefined constraints, *i.e.*, complex static ones, we propose not to use assertions, instead we use **CHECK** constraints. Also, we propose extending the use of such constraints by the possibility of the employment of many free variables (**Self**) that can or not be connected to the same table. For example, we propose to declare the constraint **{C1}** as follows:

```
ALTER TABLE SESSION
ADD Temporal CONSTRAINT Ck_C1
CHECK Self1-TEACHER.TeacherNum = Self2-
TEACHER.TeacherNum AND Self1-AUDIENCE.AudienceCod
= Self2-AUDIENCE.AudienceCod AND Self1-
GROUP.GroupCod = Self2-GROUP.GroupCod
AND NOT EXISTS
(Self1-TEACHER.TeacherNum = Self3-
TEACHER.TeacherNum AND Self1-
AUDIENCE.AudienceCod = Self3-
AUDIENCE.AudienceCod AND Self1-GROUP.GroupCod =
Self3-GROUP.GroupCod);
```

In this constraint, we used three free variables 'Self1', 'Self2' and 'Self3' for each of the three tables: TEACHER, AUDIENCE and GROUP.

C. Dynamic constraints

For the constraints of this class for which we found 'semi-graphic' expressions, it is also possible to find patterns to represent them in a simple manner. For examples:

- for the example of the constraint **{C3^d}**, we propose the pattern: **NOT DECREASE**.
- for the constraint **{C4^d}**, we propose the following pattern: **IN THIS ORDER** {'Val₁', 'Val₂', ...} [**START WITH** 'Val₁' **[OR** 'Val₂'*];

For the other dynamic constraints, which are currently represented by relational DBMS with triggers, we also propose to express them by **CHECK** constraints. The expression of these constraints requires the use of temporal logic operators. For example, we propose to declare the constraint **{C5^d}** as follows:

```
ALTER TABLE MODULE
ADD CONSTRAINT Ck_5
CHECK Self.NumTeacherResponsible
AND EXISTS
(Self-TEACHER.NumTeacher =
Self.NumTeacherResponsible
AND Self-TEACHER Since 5 Year Self);
```

D. Constraints on timestamps and on temporal instances

To express the constraints on timestamps, we use the arithmetic operators of comparison as well as Allen operators [1], *i.e.*, **BEFORE**, **AFTER**, **OVERLAPS**, **EQUAL**, **MEETS**, **DURING**, **CONTAINS**, **STARTS**, **FINISH**, etc.

In addition, to be able to express a restriction on the duration of a temporal interval, we propose the following pattern: {**MAXPERIOD** | **MINPERIOD**} Value **SCALE** [**ON** 'Val 1', 'Val 2', ...]. **SCALE** can be **YEAR**, **MONTH**, **DAY** etc.; it specifies the used temporal granule. Without the option [**ON** 'Val 1', 'Val 2', ...], the constraint is applied to all instances of the concerned element, with this option, the constraint is applied only to instances defined by the indicated values; for example, **MAXPERIOD 7 YEAR ON** 'Assistant Professor' is applied only to Assistant Professors.

For the four types of constraints on temporal instances proposed in Section 4.C.2, we propose to represent them using the following patterns:

- **VACUUMING AFTER** Value {**SCALE** | **VALUES**} [**CASCADE**]. The **CASCADE** option can progress the vacuuming for data related to the concerned data.
- **ONLY** Value {**CORRECTIONS** | **EVOLUTIONS**} **ALLOWED**.
- [**COALESED** | **NOT COALESED** [**WITH** Value **MAX VALUEEQUIVALENT**]}. By default, the instances are to keep coalesced. The option **WITH** Value **MAX VALUEEQUIVALENT** precise the maximum number of non-coalesced equivalent values.
- [**FUTURE UPDATES DESTRUCTIVE** | **FUTURE UPDATES NOT DESTRUCTIVE**].

E. Recapitulative example

We briefly illustrate here how we plan to use our proposals to enrich SQL3, through an excerpt of the command concerning the creation of the TEACHER table:

```
CREATE TABLE TEACHER V(DEFAULT, TEMPORAL
ELEMENT, DEFAULT) T
(TeacherNum NUMBER(3)
CONSTRAINT Pk_Teacher PRIMARY KEY,
LName VARCHAR2(20) NOT NULL,
Status V(DEFAULT, DEFAULT, YEAR) T VARCHAR2(2),
...
CONSTRAINT Dur_TeacherStatus Status.VT DURING
OR EQUAL TEACHER.VT,
CONSTRAINT SVT_STT_Status Status.STT <=
Status.SVT + 1 MONTH,
CONSTRAINT Vacc_TeacherStatus Status VACCUMING
AFTER 3 VALUES,
CONSTRAINT Ito_TeacherGrade Grade IN THIS ORDER
{'A', 'AP', 'Pr'} START WITH 'A' Or 'AP',
CONSTRAINT MinEs_TeacherGrade Grade MINPERIOD 3
YEARS ON 'AP',
CONSTRAINT O2E_TeacherGrade Grade ONLY 2
EVOLUTIONS ALLOWED,
CONSTRAINT Nd_TeacherSal Salary NOT
DECREASE,
CONSTRAINT Fund_TeacherSalary Salary FUTURE
UPDATES NOT DESTRUCTIVE,
CONSTRAINT Vacc_TEACHER TEACHER
VACUUMING AFTER 80 YEAR CASCADE );
```

VI. CONCLUSION AND FUTURE WORK

We presented in this paper new concepts to improve the modeling of four different kinds of facts: conventional facts (non-temporal), valid-time facts, transaction-time facts, and bitemporal facts.

Our proposals concern, first, the expression of constraints in the UML-TF profile [19], dedicated to the modeling of temporal DB. This profile allows such modeling in a simple, customizable and progressive manner, using expressive decorations. With the enrichment provided by these proposals, it is possible to take into account temporal constraints, classified into three categories: static constraints invoking a temporal dimension, dynamic constraints, and constraints on the timestamps or on temporal instances. This classification into three categories, refined if necessary on classes and sub-classes, helped us to identify appropriate ways for expressing each type of constraint. In particular, the formal expression of these constraints required the extension of OCL with new keywords.

Secondly, we have proposed enhancements to SQL3 to enable the mapping of UML-TF class diagrams in an object-relational model, attempting to take advantage of the “declarative” approach of constraints adopted by DB languages. Thus, the designer can focus his attention on the expression without worrying about the modules to be integrated into the DBMS providing their checking.

Our future work focuses on the development of new tools and the enrichment of platforms that support them in order to implement our proposals. We also plan to study how checking temporal constraints according to our proposals.

REFERENCES

- [1] Allen J. F., Maintaining Knowledge about temporal intervals, *Communication of the ACM*, 1983, pp. 832-843.
- [2] Bouaziz R. and Brahmia Z., Gestion des données temporelles dans un environnement multi-versions de schémas, *Technique et Science Informatiques*, vol. 28 n° 1, 2009, pp. 39-74.
- [3] Böhlen M. H., Valid Time Integrity Constraints, Technical Report (94-30), University of Arizona, 1994.
- [4] Böhlen M. H., Snodgrass R. T., and Soo M. D., “Coalescing in Temporal Databases.”, *Proceedings of the 22nd International Conference on Very Large DataBases (VLDB)*, Bombay, India, September 1996, pp. 180-191.
- [5] Cengarle M. V. and Knappe A., Towards OCL/RT, *Lecture Notes in Computer Science*, vol. 2391, 2002, pp. 390-409.
- [6] Cordeiro R. L. F., Edelweiss N., Galante R. M., and dos Santos C. S., “TVCL: Temporal Versioned Constraint Langage.”, *20th Simpósio Brasileiro de Bancos de Dados, Anais/Proceedings*, 2005, pp. 55-69.
- [7] Cordeiro R. L. F., Galante R. M., Edelweiss N., and dos Santos C. S., “A Deep Classification of Temporal Versioned Integrity Constraints for Designing Database Application.”, *Proceedings of the 19th International Conference on Software Engineering & Knowledge Engineering*, 2007, pp. 416-421.
- [8] Debnath N., Riesco D., Montejano G., Grumelli A., Maccio A., and Martellotto P., “Definition of new kind of UML Stereotype based on OMG Metamodel”, *Proceedings of the Arab International Conference on Computer Systems and Applications: AICCSA’03*, Tunis, 14-18 July 2003, Tunisia.
- [9] Doucet A., Fauvet M. C., Gançarski S., Jomier G., and Monties S., “Using Database Version to Implement Temporal Integrity Constraints.”, *Proceedings of the Second International Workshop on Constraint Databases*, 1997.
- [10] de Brock E. O., “A general treatment of dynamic integrity constraints”, *Data & Knowledge Engineering*, 32, 2000, pp. 223-246.
- [11] Etzion O., Jajodia S., and Sripada S. M. (Editors), *Temporal Databases: Research and Practice*, Springer-Verlag, *Lecture Notes in Computer Science*, vol. 1399, 1998.
- [12] Jensen C. S. and Snodgrass R. T., “Temporal Specialization and Generalization.”, *IEEE Transactions on Knowledge and Data Engineering*, vol. 6, n° 6, 1994, pp. 954-974.
- [13] Jensen C. S. and Dyreson C. E. (Editors), Böhlen M. H., Clifford J., Elmasri R., Gadia S. K., et al., “The Consensus Glossary of Temporal Database Concepts.”, in Etzion et al., 1998.
- [14] Jensen C. S., “Temporal Database Management.”, dr.techn. thesis by Christian S. Jensen, defended April 2000 available at: <http://www.cs.auc.dk/~csj/Thesis>.
- [15] Jensen C. S. and Snodgrass R. T. (Editors), *Temporal Database Entries* for the Springer Encyclopaedia of Database Systems, Technical Report TR-35, TIMECENTER, May, 2008.
- [16] Lomet D., Barga R., Mokbel M. F., and Shegalov G., “Transaction Time Support Inside a Database Engine.”, *Proceedings of the International Conference on Data Engineering*, 2006, pp. 35-46.
- [17] Makni A and Bouaziz R., Concurrency Control for Temporal Databases, *International Journal of Databases Management Systems*, vol. 2 n° 1, 2010, pp. 39-58.
- [18] Mkaouar M. and Bouaziz R., L’édification de framework pour l’aide au développement d’applications temporelles, *Information Science for Decision Making*, n° 21, 2005.
- [19] Mkaouar M. and Bouaziz R., UML-TF: un profil UML pour la représentation des faits temporels, *Technique et Science Informatiques*, vol. 26 n° 3-4, March-April 2007, pp. 305-338.
- [20] Oracle Corporation, *Advanced Application Developer’s Guide: Using Oracle Flashback Technology*, Oracle Documentation, 2008.
- [21] Oracle Corporation, *Workspace Manager Developer’s Guide*, Oracle Documentation, 2009.
- [22] OMG, *Object Constraint Language (OCL) Specification*, Object Management Group, www.omg.org, 2001-2010.
- [23] Roddick J. F., Schema Vacuuming in Temporal Databases, *IEEE Transactions on Knowledge and data engineering*, vol. 21 n° 5, 2009, pp. 744-747.
- [24] Snodgrass R. T., Böhlen M. H., Jensen C. S., and Steiner A., “Adding Valid Time to SQL/Temporal.”, *SQL/Temporal Change Proposal, ANSI X3H2-96-501r2, ISO / IEC / JTC1 / SC21 / WG3 DBL-MAD-146r2*, “Adding Transaction Time to SQL/Temporal.”, *SQL/Temporal Change Proposal, ANSI X3H2-96-502r2, ISO / IEC / JTC1 / SC21 / WG3 DBL-MCI-143*, 1996.
- [25] Wu Y., Jajodia S., and Wang X. S., “Temporal Database Bibliography Update.”, in Etzion et al., 1998.
- [26] Ziermann P. and Gogolla M., OCL Extended with Temporal Logic, *Lecture Notes in Computer Science*, vol. 2890, 2003, pp. 617-633.

Optimal Query Operator Materialization Strategy for Hybrid Databases

Martin Grund, Jens Krueger, Matthias Kleine, Alexander Zeier, Hasso Plattner

Hasso-Plattner-Institut

August-Bebel-Str. 88

14482 Potsdam, German

{martin.grund, jens.krueger, matthias.kleine, alexander.zeier, hasso.plattner}@hpi.uni-potsdam.de

Abstract—Recent research shows that main memory database system provide many different advantages over traditional disk based systems. Furthermore it is shown that the way how data is persisted in such a system is very important. Modern systems provide a hybrid row- and column-oriented storage layer that proves to be optimal for certain workloads. To further optimize the query execution it becomes crucial to select the best possible query operators. However, not only the implementation of the operator is very important but as well the way how intermediate results are handled. In *HYRISE*, we implemented different possibilities of query operator materialization and show in this paper when to chose which kind of output. The results of our experiments can be directly used during plan creation by a cost-based query executor.

Keywords-*Hybrid Main Memory Database; Query Execution; Column Store; Materialization.*

I. INTRODUCTION

Main memory database systems have proven to be advantageous for various scenarios ranging from high-performance analytical data warehouse accelerators to classical Online Transactional Processing (OLTP) databases. Due to the available size of main memory on a single rack server of currently 1TB almost all enterprise like applications with a mixed transactional / analytical focus can be run on such systems. Another great advantage of in-memory data processing is that data access operations are more predictable compared to disk access based operations.

Since data is no longer stored in secondary structures like the buffer pool of traditional disk based databases but operations are directly executed on the primary data it becomes important to deal with the question on how to efficiently handle intermediate results and execute any given query in the best possible way.

For our research the main focus is query execution in a hybrid main memory database system such as *HYRISE* [1]. In *HYRISE*, relational tables are partitioned into disjoint vertical partitions. Data is stored dictionary compressed and single columns can be furthermore bit-compressed to achieve higher compression ratios. In such an environment it becomes crucial to choose the right materialization strategy to lower the amount of copied data but on the other hand improve the cache miss patterns of different queries during the query plan execution.

The authors of [2] identify different materialization strategies for column-oriented DBMS and explore the trade-offs that exist between them. This paper builds on these ideas and enhances their model for hybrid main memory databases and furthermore empirically evaluates variations of them using *HYRISE*, a hybrid main memory based DBMS research prototype. Section V gives a brief summary of related work, Section II summarizes the materialization strategies presented in [2], Section III describes their adaption within *HYRISE*, and Section IV evaluates the performance of the implementation, followed by a conclusion in Section VI.

II. EXISTING MATERIALIZATION STRATEGIES

This section gives a brief summary of the materialization strategies for column-based DBMS that are identified in [2]. Compared to our extensions, their work primarily covers column-stores, while it is important for *HYRISE* to support different kinds of materialization strategies for hybrid databases. The authors recognize two different aspects of materialization strategies, time of materialization, i. e., late vs. early materialization, and parallel vs. pipelined materialization, whose influence on execution plans is explained using the following example SQL query:

```
SELECT col1, col2 FROM table
WHERE col1 < CONST1 AND col2 < CONST2
```

Abadi et al. [2] make use of specialized plan operators that are explained briefly in the following plan descriptions. The query plans are illustrated using diagrams, as the one shown in Figure 1, where *type of output* is either *mat*, *pos*, or *mat+pos*, indicating output of materialized data, positions, or materialized data and positions. Inputs that are filtered by a predicate are underlined.

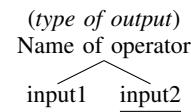


Figure 1. Example Query Plan Diagram

Table I gives a detailed list of the basic operators used during the query plans. In the following paragraphs we will present the different materialization strategies that are implemented in *HYRISE*.

Table I
DIFFERENT MATERIALIZING QUERY PLAN OPERATORS

DS1	Reads all data for a given column, applying a given selectivity. The output is a list of positions.
DS2	Reads all data for a given column, applying a given selectivity. The output is a list of position value pairs.
DS3	The data of a column is read and filtered with a list of positions. The output is a column of values corresponding to those positions.
DS4	A column is read and a list of positions is applied as a filter, tuples satisfying a predicate are selected, producing a list of positions.

Early Materialization / Pipelined: As illustrated in Figure 2, DS2 scans col1, filtering by predicate, outputting positions and data. DS4 index-scans col2 using the positions of the first scan, filtering by predicate, outputting the merged data.

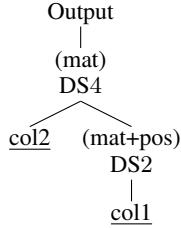


Figure 2. Plan for Early Materialization / Pipelined

Early Materialization / Parallel: As illustrated in Figure 3, SPC scans both columns in parallel, filtering by both predicates simultaneously.

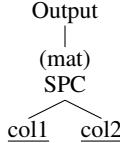


Figure 3. Plan for Early Materialization / Parallel

Late Materialization / Pipelined: As illustrated in Figure 4, DS1 scans col1, filtering by predicate, outputting positions only. DS3 and DS1 index-scan col2 using these positions, filtering by predicate, outputting positions only. Both columns are index-scanned using these positions and the results are merged.

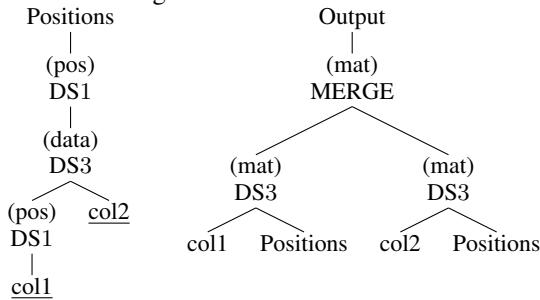


Figure 4. Plan for Late Materialization / Pipelined

Late Materialization / Parallel: As illustrated in Figure 5, each column is scanned with DS1 outputting positions. The positions are combined with an AND. These merged positions are processed as in the previous plan, i.e., both columns are index-scanned and the results are merged.

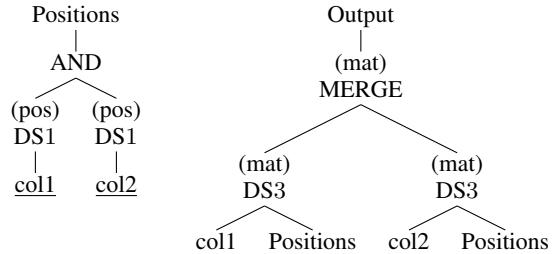


Figure 5. Plan for Late Materialization / Parallel

III. ADOPTION IN HYRISE

This section describes how the example query plans were adapted in *HYRISE*.

A. Plan Operators

The implementation makes use of new plan operators that are extensions of existing *HYRISE* plan operators. They do not directly correspond to the plan operators introduced in Section II but are an adaption of them to *HYRISE*.

TableScan: The *TableScan* provided by *HYRISE* scans all columns of a table in parallel, applies predicates, and outputs materialized data. Input can be either a list of positions or materialized data.

PositionTableScan: The *PositionTableScan* scans all columns of a table in parallel, applies predicates, and outputs positions. It corresponds to the DS1 operator.

MaterializingScan: This new plan operator is based on the DS2-operator of [2]. It accepts a raw table or a table with associated positions as input. In addition, it accepts predicates. As DS2, it produces materialized data and positions. Being a projection scan, it produces a new table that contains a configurable subset of the columns of the original table. It additionally outputs a list of positions that indexes the original table.

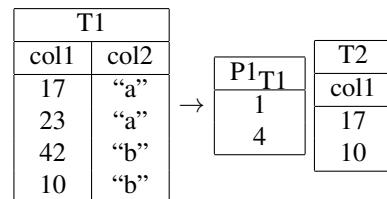
Figure 6. MaterializingScan on an example table T1 with predicate $\text{col1} < 20$ produces positions $P1_{T1}$ and materialized result T2

Figure 6 shows an example application of this plan operator. A *MaterializingScan* is applied to table T1. No extra positions are given as input. The predicate supplied is $\text{col1} < 20$ and col1 is the only column to be projected. The result is a new position list as well as a new table with the filtered column col1 .

TableScanUsingExistingData: This new plan operator is based on the DS4-operator of [2]. It is designed to work on the output of a *MaterializingScan*. As input it takes a table, a list of positions into this table as well as a materialized version of some columns of the table at

these positions. It also accepts predicates. It index-scans the input table and outputs a materialized table. The materialized columns that are input into this operator are not directly used as output. Instead only the rows where the predicates match are copied into the output. Thus, the operator produces a table with the same layout as the first input table.

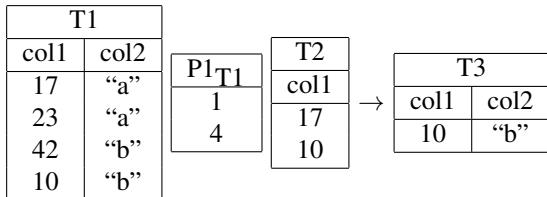


Figure 7. TableScanUsingExistingData on the results of the last example, with the predicate $\text{col2} = \text{"b"}$

Figure 7 shows the application of an TableScanUsingExistingData to the results of the example shown in Figure 6, using the predicate $\text{col2} = \text{"b"}$. T1 is scanned using the position of $P1_{T1}$. For all rows where col2 matches the predicate, the data from col2 of T1 and col1 of T2 is added to the result.

B. Query Plans

The query plans introduced in [2] have been adapted for HYRISE. As they are not exactly the same plans, they are given different names to avoid confusion.

Plan 1 - One Scan: This plan corresponds to the early materialization / parallel scan. As illustrated in Figure 8, it consists of only one plan operator that accesses both input columns simultaneously, i. e., for each row both columns are read and written to the output if both predicates match.

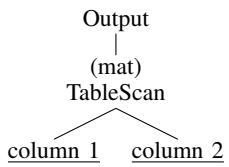


Figure 8. Plan 1 - One Scan

Given a column layout, this plan is expected to be efficient if the number of values that have to be accessed is high for both columns. It is expected to be relatively inefficient if the selectivity is low on column 1 as in that case both columns will still always be read. That is not the case for the other plans.

Given a row layout, this plan is expected to be efficient for low to high selectivities as long as the columns are adjacent or not too wide, as then reading both columns simultaneously causes less cache misses than reading them sequentially. For very low selectivities, the position based scans might still be more efficient, as then the number of total cache accesses is expected to be much lower for them than for the *One Scan* plan, even if the number of cache misses is expected to be slightly higher.

Plan 2 - No Data: This plan has no direct correspondence to the original execution plans, but it is an optimized version of the late materialization / parallel plan. As illustrated in Figure 9, it consists of two plan operators. A PositionTableScan on column 1, which produces only positions for the rows where the predicate on column 1 matches. This operator is followed by a TableScan, which filters column 2 by its predicate at these positions and materializes both columns.

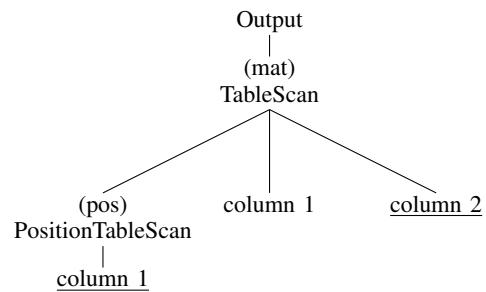


Figure 9. Plan 2 - No Data

Given a column layout, this plan is expected to be efficient when selectivity is low on column 1. Then, unlike in the *One Scan* plan, not many of the second column's rows have to be accessed. The plan is expected to be less efficient than the *One Scan* plan if the selectivity is high on column 1 and low on column 2, as then both columns have to be read nearly completely in both plans, but this plan generates a large amount of unused temporary position data.

Given a row layout, this plan is expected to be worse than the *One Scan* plan for most of the selectivities, especially if the columns are adjacent or are not too wide, so that one row of both columns fits into one cache line. For very low column 1 selectivities though, this plan is expected to require roughly half of *One Scan*'s cache accesses with only slightly higher cache misses, and might thus be faster.

Plan 3 - Data: This plan is closest to the Early materialization / pipelined plan. As illustrated in Figure 10, the MaterializingScan produces positions as well as values for column 1 where the predicate matches. Unlike the DS2 operator it produces these as two separate columns, not one column containing (position, data) pairs, the reason being a HYRISE implementation detail. The TableScanUsingExistingData scans column 2 at these positions. The rows where the second predicate matches are output materialized.

Given a column layout, this plan is expected to be more efficient than the *No Data* plan for very low selectivities on column 1. Then, the materialization of the final result, i. e., the SimpleTableScan in the *No Data* or the SimpleTableScanUsingExistingData in this plan, has to access very few rows of column 1. Doing so by accessing the original column by index is more expensive than sequentially accessing pre-materialized data. It is expected to be inefficient if selectivity

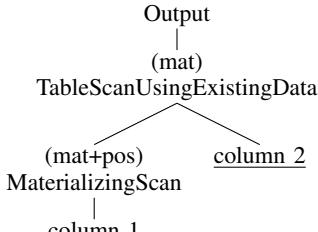


Figure 10. Plan 3 - Data

is high on column 1 and low on column 2, as a large part of column 1 will be materialized that will later on not be used. Given a row layout, the expectations are similar to those of the *No Data* plan, with the *Data* plan again performing better for lower selectivities than the *No Data* plan.

IV. EVALUATION

As already pointed out in Section III-B, each of the plans is expected to work best at a certain configuration of selectivities. In order to empirically evaluate these assumptions, the algorithms were run on an IBM Series Blade, Xeon 5450, 64 GB RAM using different table layouts. First, a 2-column table stored in column-layout. Second, a 2-column table stored in row-layout. Third, a 60-column table stored in row-layout. Each table contains 1.000.000 rows; each column is 4 byte wide. The data was generated using the *HYRISE* data generation tool.

A. Column Layout

In order to measure the general performance, each plan was run for selectivities varying for both columns, each selectivity ranging from 0 to 1 in steps of 0.01. For each pair of selectivities, each plan was run and the total CPU cycle count was measured and averaged over three runs.

Figure 11a shows the algorithm with the lowest total CPU cycle count for each combination of selectivities. As can be clearly seen, the influence of the first column's selectivity is considerably larger than that of the second. As expected, the *One Scan* plan outperforms the other plans for high selectivities on both columns whereas the position based scans are better at lower selectivities on column 1.

Figure 11b shows the ratio of CPU cycles of the algorithm with the highest count to that with the lowest count at the given selectivities. This ratio ranges from values between approximately 1 and 2.4. Two areas can be identified where high ratios appear. First, for a low selectivity on column 1 independent of the second column's selectivity. Second, for a high selectivity on column 1 and a low selectivity on column 2.

The relative performance at these extremes can be seen in more detail in Figure 12a, which shows the CPU cycles of all three plans for two fixed selectivities for column 2 of $1.25e-3 = (25 \text{ rows} / 2.000.000 \text{ rows})$ and 1.0 in dependence of the first column's selectivity. Figure 12b depicts the same

using a logarithmic scale on the x-axis, allowing differences for low selectivities on column 1 to be discerned more easily.

For low selectivities on column 1 the ratio from *One Scan* to *Data*, i. e., the highest to the lowest CPU count, is close to 2. This is expected behavior, as the position based algorithms do not have to read much of column 2 whereas the *One Scan* algorithm always must read both columns.

As can be seen in Figure 12a, the ratio of *One Scan* to *Data* is largest for a high selectivity on column 1 and a low selectivity on column 2. As can be seen in Figure 12c, *Data* also accesses twice as much memory as *One Scan*. This is expected. For this configuration, *One Scan* reads both input columns once and writes nothing, whereas *Data* reads the first column, writes positions and materialized data and reads the second column, thus performing roughly twice as many memory accesses as the *One Scan* algorithm.

Given a high selectivity on both columns, *One Scan* is only 1.25 times faster than *Data*, as can be seen in Figure 12b, and requires 1.4 times the number of L1 cache accesses. We measured, that for this configuration of selectivities about a third of *Data*'s CPU cycles is used by the *MaterializingScan* while the remaining cycles are used by the *TableScanUsingExistingData*, making the *TableScanUsingExistingData* 1.2 times faster than the complete *One Scan*. This is interesting, as the scan using existing data has to perform more work than the *One Scan* plan. While the *One Scan* plan reads and writes both columns, the *TableScanUsingExistingData* does the same but additionally reads positions.

For this configuration of selectivities, further investigation is required to identify the reasons for the unexpectedly good performance of the *Data* plan. Nevertheless, for the most configurations of selectivities the plans perform as expected.

B. Row Layout

The queries were executed on a 2-column and a 60-column table. Figure 13 shows the CPU cycles for fixed column 2 selectivities for the 2-column layout. As can be seen, the performance is very close to the column-layout performance. The *One Scan* plan is, as expected, faster than the others for high selectivities, as it only has to read one continuous block of data once whereas the position based scans have to process this block twice, thus causing more cache misses. As can be seen in Figure 13a, *One Scan* is still slower than the position based scans for low column 1 selectivities. There, the number of cache accesses that were measured for the *One Scan* plan, which evaluates predicates on both columns in parallel, were, as expected, higher than for the other plans, with the cache misses measured only slightly higher.

In order to analyze the algorithms' performance for larger containers, they were executed on a 60-column / 1 container table. All queries still output only the first two columns. The *MaterializingScan* of the *Data* plan still only

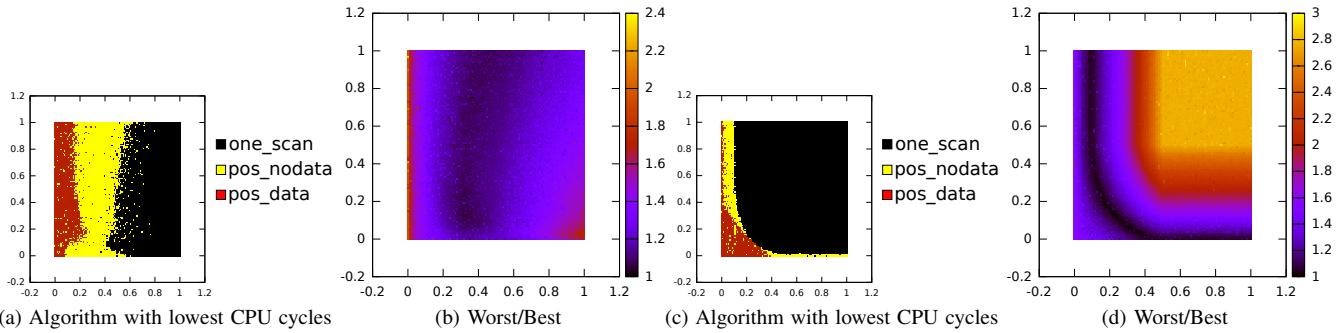


Figure 11. Figure (a) and (b) for 2 columns / 2 containers and (c) and (d) for 60 columns / 1 container. Comparing CPU cycles of all three algorithm across selectivities; x-axis = Column 1; y-axis = Column 2

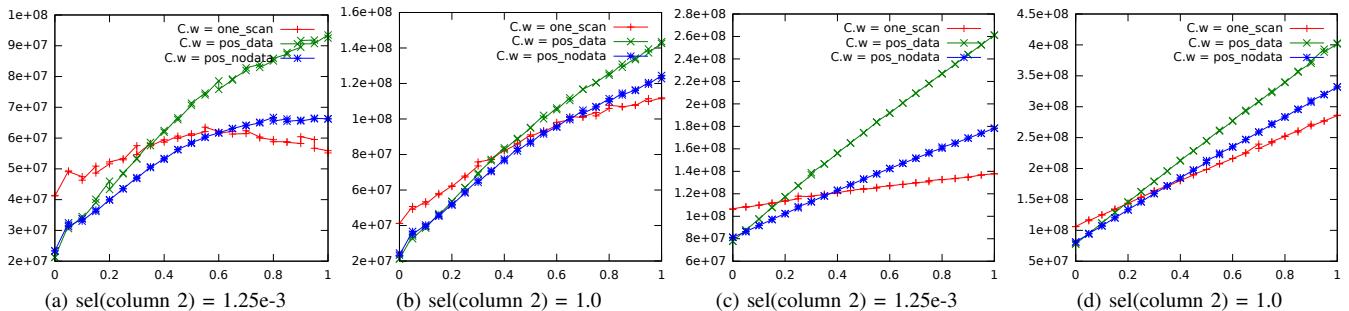


Figure 12. 2 columns / 2 containers. 12a and 12b Total CPU cycles. x-axis = selectivity on first column; Figure 12c and 12d are L1 data cache accesses for the same experiment. Selectivity on second column is fixed for all graphs.

materializes the first column. Figures 11c and Figure 11d give a general overview of the algorithms' performance. Unlike in previous benchmarks, the performance is strongly influenced by the selectivity of both columns. The *One Scan* plan performs best if the combined selectivity is not low and it is roughly 2.8 times faster than the *Data* scan for a selectivity of 1 on both columns. If selectivity is low on column 1 or very low on column 2, the position based plans are best, the *Data* plan performing roughly 1.6 times faster than the *One Scan* plan for very low selectivities on both columns.

Figure 13d provides a snapshot at fixed column 2 selectivities. Given a high column 1 selectivity, the *One Scan* plan surprisingly is slower for low column 2 selectivities than it is for high column 2 selectivities. This is the case even though the number of cache misses and cache accesses is lower for low column 2 selectivities than it is for high ones. Other counters, such as the number of branch mispredictions have not yet been measured and so a clear assessment can not be made, yet.

V. RELATED WORK

The topic of materialization strategies has already been researched in the context of column-based as well as row-based DBMS. Abadi et al. [3] provide a general overview of column- and row stores and identifies the importance

of late materialization in column stores. In [2], [4] Abadi et al. provide an evaluation and comparison of different materialization strategies in column-based DBMS. Different materialization strategies are identified and their performance for different kinds of queries is evaluated.

Ivanova et al. [5] analyze how materialized query plan results can be cached and reused for future queries to reduce execution times. This aspect of materialization strategies is complementary to the ones analyzed in this paper.

The materialization strategies that are analyzed in this paper are implemented in a operator at a time query execution engine. Zukowski et al. [6] follow a different approach by materializing vertical data fragments at a time, trying to restrict the data to the CPU cache.

In addition, the implementation of compression for such main memory databases becomes more and more important as shown in [7], [8]. Krueger et al. show in [9] that it is possible to further optimize read optimized column databases with compression to possibly satisfy OLTP workloads.

VI. CONCLUSION

As has been seen in IV, the materialization strategies that were introduced in III mostly exhibit the expected relative performance for the simple selection query, especially if data is stored in column-layout. Table II summarizes the best and worst performance of each algorithm in dependence of the

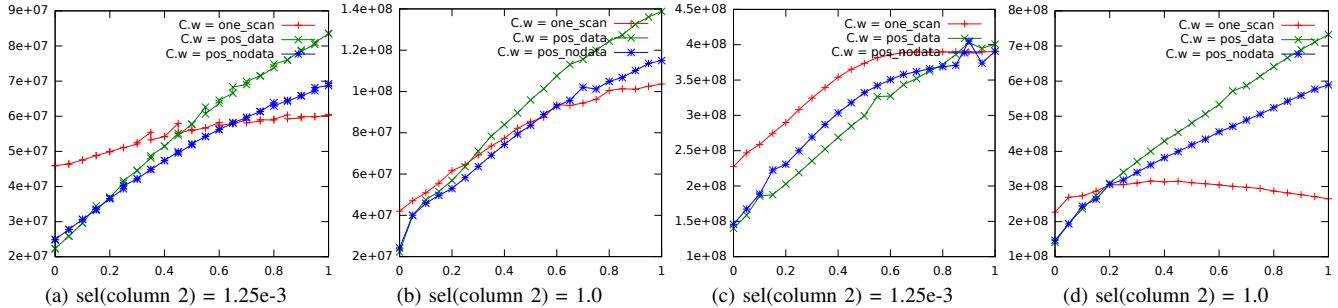


Figure 13. 2 columns / 1 container for Figure 13a and 13b and 60 columns / 1 container for Figure 13c and 13d. Total CPU cycles. x-axis = selectivity on first column. Selectivity on second column is fixed

columns selectivity. As can be seen, the best plan depends mainly on the first column's selectivity.

Table II

SELECTIVITY CONFIGURATIONS THAT DELIVER THE BEST / WORST RESULTS FOR A COLUMN-LAYOUT

Plan	Best		Worst	
	Col 1	Col 2	Col 1	Col 2
One scan	high	high	low	-
No Data	low	-	high	low
Data	very low	-	high	low

The performance for multi-column containers did not completely match our expectations, with the *One Scan* plan performing unexpectedly slow for a high selectivity on column 1 and a low selectivity on column 1. As expected though, the *One Scan* plan performs best for most of the configurations, with the position based plans providing better performance when selectivity is low on column 1 or very low on column 1.

All together, the choice of materialization strategy greatly influences the query execution time. For the simple selection query analyzed, choosing the wrong materialization strategy resulted in up to 2.8-fold increased CPU cycles. For the simple example, the optimal materialization strategy can be predetermined if the table layout is known and the selectivities of the predicates can be estimated in advance. This can be achieved by applying established techniques that rely on collecting statistical data, such as histograms [10].

An actual implementation that chooses materialization strategies automatically for container-based DBMS, such as *HYRISE*, requires further research. The analysis performed on multi-column containers has shown that the layout of tables has a strong influence on materialization strategies. Yet, aspects such as the positions of columns within containers have not been analyzed, so that further research is required to turn these observations into knowledge that can be applied at query plan construction time.

Apart from that, many other areas have been left untouched, such as analyzing different kinds of queries or tak-

ing different types of data storage, e. g., compressed position lists, into account. Still, this paper and the accompanying implementation provide the required knowledge to leverage different materialization strategies in a hybrid main memory database system such as *HYRISE*.

REFERENCES

- [1] M. Grund, J. Krüger, H. Plattner, A. Zeier, P. Cudré-Mauroux, and S. Madden, "Hyrise - a main memory hybrid storage engine," *PVLDB*, vol. 4, no. 2, pp. 105–116, 2010.
- [2] D. Abadi, D. Myers, D. DeWitt, and S. Madden, "Materialization Strategies in a Column-Oriented DBMS," in *ICDE 2007*, pp. 466–475.
- [3] D. J. Abadi, S. Madden, and N. Hachem, "Column-stores vs. row-stores: how different are they really?" in *SIGMOD Conference*, 2008, pp. 967–980.
- [4] D. Abadi, "Query execution in column-oriented database systems," Ph.D. dissertation, Massachusetts Institute of Technology, 2008.
- [5] M. Ivanova, M. L. Kersten, N. J. Nes, and R. Goncalves, "An architecture for recycling intermediates in a column-store," in *SIGMOD Conference*, 2009, pp. 309–320.
- [6] M. Zukowski, P. Boncz, N. Nes, and S. Heman, "MonetDB/X100-a DBMS in the CPU cache," *Data Engineering*, vol. 1001, p. 17, 2005.
- [7] T. Westmann, D. Kossmann, S. Helmer, and G. Moerkotte, "The implementation and performance of compressed databases," *SIGMOD Record*, vol. 29, no. 3, pp. 55–67, 2000.
- [8] D. J. Abadi, S. Madden, and M. Ferreira, "Integrating compression and execution in column-oriented database systems," in *SIGMOD Conference*, 2006, pp. 671–682.
- [9] J. Krüger, M. Grund, C. Tinnefeld, H. Plattner, A. Zeier, and F. Faerber, "Optimizing write performance for read optimized databases," in *DASFAA (2)*, 2010, pp. 291–305.
- [10] V. Poosala, P. Haas, Y. Ioannidis, and E. Shekita, "Improved histograms for selectivity estimation of range predicates," *ACM SIGMOD Record*, vol. 25, no. 2, p. 305, 1996.

From Synchronous Corpus to Monitoring Corpus, LIVAC: The Chinese Case

Benjamin K. Tsou Andy C. Chin
Research Centre on Linguistics &
Language Information Sciences
The Hong Kong Institute of Education
Tai Po, Hong Kong
btsov@ied.edu.hk andychin@ied.edu.hk

Oi Yee Kwong
Department of Chinese, Translation & Linguistics
City University of Hong Kong
Kowloon Tong, Hong Kong
olivia.kwong@cityu.edu.hk

Abstract—Very large corpora of properly processed textual materials are uncommon but they can provide important resources for language modeling in natural language processing, ranging from speech processing and text input to automatic IR and patent translation. However, when properly cultivated in spatial-temporal terms, they can foster innovative knowledge discovery in database applications by functioning as monitoring corpus and enhance the human centered communication environment by allowing more substantive introspection and comparison of linguistic and social-cultural developments of the relevant speech communities.

This paper discusses how the gigantic synchronous and homothematic corpus of Chinese, LIVAC, can contribute to the monitoring the linguistic homogeneity and heterogeneity diachronically and synchronically. After processing media texts of more than 400 million Chinese characters over 16 years, LIVAC has yielded a lexical corpus of 1.5 million words. This paper examines some aspects of the nature and extent of lexical and morphological divergence and convergence in the Chinese language of Hong Kong, Taipei and Beijing. Additional discussions cover creation and relexification of neologisms, categorial fluidity and the associated challenges to terminology standardization, such as renditions of non-Chinese personal names. This paper also explores how the associated socio-cultural developments can be fruitfully monitored by means of this unique spatial-temporal corpus.

Keywords- monitoring corpus; synchronous corpus; homothematic coprus; LIVAC; the Chinese language

I. INTRODUCTION

Although Chinese is the native or official language in many communities such as Mainland China, Taipei, and Hong Kong, its homogeneity cannot be simplistically assumed. In fact, there are readily noticeable and significant linguistic differences among these Chinese speech communities as any casual newspaper reader from a community other than his or hers will readily testify. This phenomenon can be well illustrated by the lexical items. As a consequence of recent history and localized cultural developments, the differences are arguably much greater than those among British English, American English and Australian English if Chinese-English bilinguals have an opportunity to reflect on the two situations. These linguistic differences are not only significant for NLP and linguistic

analysis but for monitoring the speech communities in which these linguistic variations are embedded.

II. USING A SYNCHRONOUS & HOMOTHEMATIC CORPUS FOR MONITORING CHINESE LANGUAGE DEVELOPMENT

In order to explore the significance of the non-homogeneity of the Chinese language in different Chinese speech communities, this paper attempts to exploit a viable and rigorous methodology which can provide, among other things, a useful foundation for research into terminology and standardization of the language.

The use of corpus has been a major means for studying natural language in authentic use rather than in abstraction [1] [2] [3]. There is now an over-abundance of natural language data for constructing linguistic corpora. However, it is important to control the nature, size and time frame of these sources when building corpora, especially when we need to conduct synchronic and/or diachronic comparisons.

Internet has become a major source for obtaining linguistic data because it is easily and readily accessible. However, we have to be cautious when drawing data from the Internet. One commonly seen phenomenon is data duplication where the same data with exact wordings and layout appear more than once on the Internet. Moreover, the timeframes of the data obtained from Internet are neither specified nor easily controlled. Overlooking these problems will lead to serious faults in drawing conclusion, especially when qualitative conclusions are based on the quantitative analysis of these data. Thus it is important to control the data rigorously in terms of both dimensions of time and content.

One major approach in corpus linguistic research is using *balanced corpus* in which data of a language are drawn from a wide range of sources/register. Examples in the English language include the British National Corpus (BNC) and American National Corpus (ANC). This type of corpus provides a comprehensive overview of the language of a particular community, such as British English and American English. It cannot however compare the same type of language in both spatial and temporal dimensions.

In this paper, we argue that heterogeneity rather than homogeneity should be assumed in the Chinese language, both lexically and syntactically, across some major Chinese communities, such as Beijing, Hong Kong, and Taipei. The LIVAC corpus [4] initially developed at the Language Information Sciences Research Center at the City University

of Hong Kong since 1995 is particularly suited for this kind of study. LIVAC is synchronous and homothematic in nature, which rigorously and regularly draws comparable amount of data from similar sections such as front page, financial page, Cross-Straits news page, editorial page, entertainment page, sports page, and local news page, of printed Chinese media of major Chinese communities (see Table I) [5] [6]. In other words, the data are analyzed within the same framework in terms of size, time, domain as well as content across communities and this provides a common platform for meaningful synchronic and/or diachronic comparisons [7]. This “Windows” approach thus ensures that comparable data are extracted according to the same set of criteria [8].

The use of massive news media materials for such a study is very much justified because the popular media should reflect the language and the readership of society and be responsive to their language preference [9] [10] [11] [12]. Moreover, such a database facilitates higher order knowledge discovery and the analysis of associated linguistic characteristics with the larger context of its human users.

Currently LIVAC has obtained 1.5 million word types by accumulatively analyzing over 400 million Chinese characters of newspaper texts in major Chinese communities.

¹ Background details of LIVAC are summarized in Table I.

TABLE I. SUMMARY HIGHLIGHTS OF LIVAC CORPUS

Communities covered:	Beijing, Hong Kong, Macau, Shanghai, Singapore, Taiwan, Shenzhen, Zhuhai, Guangzhou
Source of data:	Representative newspapers from each community
Time span:	Since 1995 (i.e., 16 years)
Coverage:	News sections: International, editorials, Cross-Straits, local, financial, entertainment, sports, etc.
Size of corpus:	1.5 million word types culled from 400+ million Chinese character of texts

With the Windows approach described above, the data of Hong Kong, Taipei and Beijing are rigorously processed and are normalized in terms of size and timeframes which can allow us to observe the differential trends of Chinese language and related linguistic developments.

III. LEXICAL CONVERGENCE AND ACTIVE-CORE VOCABULARY

Even though Hong Kong, Taipei and Beijing share the Chinese language, there are significant differences among their large lexical databases. Table II indicates the extent of lexical items shared by the three communities between 1995 and 2004, based on the above Windows approach.

TABLE II. OVERLAPPING LEXICAL ITEMS IN HONG KONG, TAIPEI AND BEIJING BETWEEN 1995 AND 2004

%	Hong Kong	Taipei	Beijing
Hong Kong		39	34
Taipei	41.1		33.6
Beijing	41.1	38.6	

¹ All the texts have been automatically segmented with semi-automatic verification, and large sections have been POS-tagged and verified. Some details on the relevant information mining efforts have been reported in [5] and [6].

The corresponding percentages between any two communities are not necessarily identical because the total numbers of words in each community are different. Consider Hong Kong and Taipei, 39% of the 215k words from Hong Kong can be found in Taipei while 41.1% of the 205k words in Taipei appear in Hong Kong. These figures show that the number of lexical items actively shared by any two of these three communities over the 9-year period is not very high. Generally speaking, less than half of the lexical items used in any one community can be found in the other two communities. The extent of overlap between Taipei and Beijing is the least. Only 33.6% of such overlapping items are found in the Taipei corpus. This demonstrates that between Taipei and Hong Kong, as well as between Hong Kong and Beijing, there are more extensive overlaps than between Beijing and Taipei. This situation might reflect the social, cultural situation associated with real politics.

In this 9-year period, over 56,000 lexical items were found in common in the three communities (see Table III).

TABLE III. OVERLAPPING LEXICAL ITEMS AMONG HONG KONG, TAIPEI AND BEIJING (1995-2004)

No. of overlapping items	Hong Kong	Taipei	Beijing
56693	26.3%	27.7%	31.8%

The extent of pairwise overlap between the communities is given in Figure 1. One may find it surprising that the extent of overlap is exceptionally low even though the three communities share the same language. Table III shows that such a core vocabulary only accounts for from 26% to 31% of the total items used in each community. It should also be noted that the same concept is not necessarily rendered by the same lexical item in the three communities. This is a partial reason leading to the low degree of lexical overlap. Therefore, lexical variation cannot be simply studied on a quantitative basis and a systematic qualitative investigation has to be carried out in order to compare lexical divergence across communities. This will be discussed in Section IV.

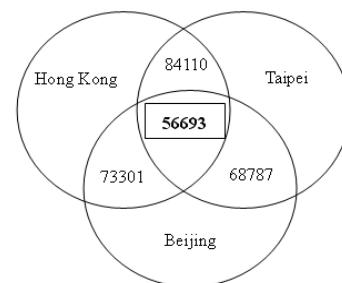


Figure 1. Extent of Lexical Overlap in Hong Kong, Taipei and Beijing in LIVAC (1995 – 2004)

These overlapping words can be considered the **active-core vocabulary** that is in current use in the language. Those non-overlap words can be considered **ambient vocabulary**, which can be divided into two sub-types:

(a) **Transparent and readily decodable**: Even though the three communities use Chinese characters to coin words, mutual intelligibility cannot be always assumed. For

example, 計程車 (*ji-cheng-che*, count-distance-car) and 出租車 (*chu-zu-che*, hired-car) are used to refer to “taxi” in Taipei and Beijing respectively, members of each community should be able to understand the other word by adding up the meaning of each morpheme (i.e., 計(count), 程(distance), 出租(hire) and 車(car)). These two words can thus be considered *mutually intelligible* between the two communities, as well as in other Chinese communities such as Hong Kong and Singapore where 的士 (*di-shi*) and 德士 (*de-shi*) are used respectively. Another pair of example is 軟盤 (*ruan-pan*, soft-platter) and 軟碟 (*ruan-die*, soft plate) for *floppy disc*: The former is used in Mainland China and Taipei, while the latter in Hong Kong. 盤 and 碟 are similar in meanings and people should not find problems in understanding the alternate term.

(b) **Opaque and non-readily decodable:** Some words are less mutually intelligible across communities. For example, 的士 (*di-shi*) and 德士 (*de-shi*) meaning “taxi” used in Hong Kong and Singapore respectively cannot be simply derived from the meanings of their components (i.e., 的(*di*), 士(*shi*) and 德(*de*)) because these two words are created by means of phonetic adaptation. They are thus less mutually intelligible to members of Beijing and Taipei.

It is notable that beyond the Chinese context, there is also a considerable degree of mutual intelligibility between words used in Chinese and Japanese (i.e., those written with Japanese kanji) on vehicle-related words. Chinese readers are found to be able to understand more vehicle-related words written with Japanese kanji than vice versa [13]. It is also noted that the extent of overall mutual intelligibility for understanding Chinese items by Japanese has decreased from 51% to 25% when the same window is taken 10 years later [14] and this deserves fuller investigation. One possible reason for the decrease is that many VEHICLE words in Chinese are phonetically adapted (mostly from English), such as *ji-pu-che* 吉普車 (jeep), *mo-tuo* 摩托 (motorbike). The meanings of these words are not transparent from the Chinese characters, i.e., the meaning of 摩托 is not simply a combination of the meanings of 摩 and 托.

Although 的士 (*di-shi*), 計程車 (*ji-cheng-che*, count-distance-car) and 出租車 (*chu-zu-che*, hired-car) appear in all three communities, their frequency distributions are significantly different across the three communities: 的士, 計程車 and 出租車 are predominantly used in Hong Kong, Taipei and Beijing respectively. It is thus more appropriate to consider the other two items ambient vocabulary in each community. In this regard, we re-define *ambient vocabulary* as those items whose frequency in a particular community accounts for 80% or above of the total frequencies from all the three communities. Table IV provides the quantitative data of this type of vocabulary in the three communities.

TABLE IV. WORDS WITH OVER 80% LOCAL USAGE FREQUENCY

%	Hong Kong	Beijing	Taipei
Type	55.8	51.8	59.5
Token	12.8	9.9	11.7

The data show that about half of the lexical items of each community have high local usage frequency. In terms of tokens, these local high frequency items only account for around 10% of the overall token usage. This again demonstrates that there is a high degree of lexical heterogeneity across these three Chinese communities.

The above discussion draws attention to the considerable heterogeneity of the Chinese language among Hong Kong, Taipei and Beijing. Furthermore, we should also point out that non-reciprocal items found in one single community will subsequently spread to other communities upon frequent cross-communal contact and will gradually become the active-core vocabulary. This dynamic nature of lexical development will be explored in the next section.

IV. RENDITIONS OF FOREIGN PERSONAL NAMES

The tremendous growth and attrition of proper names in the Chinese language has become a challenge in NLP, especially for named-entity recognition [15] [16]. Chinese, unlike English, does not have any means, such as capital letters to identify proper names. Thus, in LIVAC, three types of proper names (personal names, geographical names and organization names) are separately tagged in multiple ways.

While phonetic adaptation is commonly used to render foreign personal names in Chinese, the three communities show considerable variations which are critical to NLP in a cross-linguistic context. Such differences can be attributed to the use of different dialects for the transliteration template. For example, Cantonese is the local dialect providing the usual base for phoneticization in Hong Kong while Mandarin is the base for Beijing and Taipei. Furthermore, even for those popular figures whose names appear frequently in news media, discrepancies across communities also exist so that members from one community may not recognize readily that two different Chinese renditions in fact could refer to the same individual [17] [18]. Table V lists some well-known non-Chinese names rendered differently in the three communities, according to LIVAC.

TABLE V. NON-CHINESE PERSONAL NAMES WITH DIFFERENT RENDITIONS IN LIVAC

Names	Hong Kong	Taipei	Beijing
George W. Bush	布殊	布希	布什
Tony Blair	貝理雅	布萊爾	布萊爾
Saddam Hussein	薩達姆	哈珊	薩達姆
Zinedine Zidane	施丹	席丹	齊達內
Whoopi Goldberg	胡比高拔	琥碧戈柏	烏比·戈德堡
Brad Pitt	畢彼特	布萊德彼特	布拉德皮特

Besides the dialects involved in the recipient language, there are other communal differences, such as the number of syllables in the transliteration, as shown by the renditions of Brad Pitt and Whoopi Goldberg. Furthermore, even within the same community, different domains might have different principles for transliteration. For example, in the domain of entertainment, both first name and last name are always included in the transliteration, while in the political and sports domains, only the last names are transliterated.

V. RELEXIFICATION

The lexical divergence across Chinese communities can be reduced through **relexification** [8]. In the initial stage, there can be alternate lexical items referring to the same concept in different communities. Subsequently, these lexical variants compete among each other and some are retained and become core items. *Internet* and *mobile phone* are good examples to illustrate the relexification process.

A. Internet

The rapid developments in computer technology have led to the coinage of new words. The lexical variation in the IT domain can be best illustrated by those words designating *Internet*. In LIVAC, there have been at least 13 lexical items referring to this technology since it was first introduced, as shown in Table VI below.

TABLE VI. ALTERNATE RENDITIONS OF “INTERNET” IN LIVAC

1. 互聯網 mutual-link-net	8. 網際網絡 inter-net-network
2. 互聯網絡 mutual-link-network	9. 網際網路 inter-net-network
3. 交互網 cross-mutual-net	10. 遞訊網 transmit-information-net
4. 信息網 information-net	11. 英特網 INTER-net
5. 訊息網 information-net	12. 因特網 INTER-net
6. 國際網 international-net	13. 萬維網 10K-dimension-net
7. 國際聯網 international-link-net	

The data show that when *Internet* was first introduced, there were diverse renditions for this technology in the Chinese communities. Items 1 – 10 are created by means of semantic adaptation with the functions and characteristics of *Internet* being described. Items 11 - 13 are created by means of phonetic adaptation or hybrid (a combination of both semantic and phonetic adaptations) by which the pronunciation of “internet” in English is modeled. It is interesting to observe that after subsequent relexification and merger, 互聯網 (mutual-link-net) became the most popular term with 因特網 (INTER-net) as the next frequently used item by year 2000 (for more details on Chinese neologicistic development, see [7], [19] and [20]).

B. Mobile phone

The LIVAC data point to at least 10 items referring to *mobile phone* in Chinese, as shown in Table VII below:

TABLE VII. ALTERNATE RENDITIONS OF “MOBILE PHONE” IN LIVAC

手持電話 hand-hold-phone	無線電話 no-wire-phone
手提電話 hand-carry-phone	隨身電話 follow-body-phone
行動電話 action-phone	攜帶電話 carry-phone
流動電話 transient-phone	大哥大 Big-Boss-Brother
移動電話 mobile-phone	手機 Hand-phone

We find significant convergent developments in Hong Kong, Taipei and Beijing, and discrete changes in the choice among alternate forms when comparison is made on with three consecutive annual windows (from 1998 to 2001), as shown in Table VIII below:

TABLE VIII. DEVELOPMENT OF LEXICAL ITEMS RELATED TO “MOBILE PHONE” FROM 1998 TO 2001

Years	Hong Kong	Taipei	Beijing
98-99	手提電話 hand-carry-phone	行動電話 action-phone	移動電話 mobile-phone
	流動電話 transit-phone	大哥大 Big-Boss-Brother	大哥大 Big-Boss-Brother
99-00	流動電話 transit-phone	行動電話 action-phone	手機 hand-phone
	手機 hand-phone	手機 hand-phone	移動電話 mobile-phone
00-01	流動電話 transit-phone	手機 hand-phone	手機 hand-phone
	手機 hand-phone	--	行動電話 action-phone

The three communities initially had different neologicistic renditions for *mobile phone*. 手提電話 (*shou-ti-dian-hua*, hand-carry-phone), 行動電話 (*xing-dong-dian-hua*, action-phone) and 移動電話 (*yi-dong-dian-hua*, mobile-phone) were used most frequently in Hong Kong, Taipei and Beijing respectively. In 1999-2000, the disyllabic item 手機 (*shou-ji*, hand-phone) was the most frequently used in Beijing while it was the next frequently used item in Hong Kong and Taipei. In 2000-2001, it completely took over other items and became the core item for *mobile phone* in all three communities. There can be a number of reasons for one item winning over the others and disyllabification could be one of such reasons since it is the major trend of lexical development in the Chinese language. According to Masini’s study, the ratio between monosyllabic and polysyllabic words is approximately 1:6. Among these polysyllabic words, over 70% are disyllabic [21]. In her sociolinguistic study on the nature of “Chinese word” [22], Wang found that over 90% of the words segmented by her informants are disyllabic. The propensity for disyllabification has often been noted [23] [24] [25] [26].

VI. CATEGORIAL FLUIDITY IN CHINESE

Chinese is an isolating language which lacks morphological markings to distinguish different parts-of-speech (POS). For example, the word 懷疑 (*huai-yi*), with a verb sense (“to suspect”) to start with, should only appear as a verb in a dictionary, despite its variable usages found in real contexts, as in (a) – (c) below.

- (a) 我 懷 疑 他 是 賊
wo huai-yi ta shi zei
‘I suspect he is a thief’
- (b) 他 滿 臉 懹 疑 表 情
ta man-lian huai-yi biao-qing
‘He wears a suspicious look’
- (c) 這 只 是 我 的 懹 疑
zhe zhi shi wo de huai-yi
‘This is only my suspicion’

In (a), 懹疑 (*huai-yi*) is a verb. In (b), it is an adjective and in (c), it is a noun.

We call this relative flexibility of a word being used for different grammatical functions and possibly different POSs *categorial fluidity* [27]. In the following, we only focus on the fluidity between verbs and nouns. We consider categorial fluidity a continuum. We will show, by means of the LIVAC data, how categorial shift takes places across Chinese communities and over time.

A. Methodology

First, all verbs (excluding copula verbs and auxiliary verbs) with their frequencies were extracted. Then out of these verbs, those which also exhibited noun or nominalized usages were extracted together with the corresponding frequencies. We call this set of verbs “VN words”.

Next, a simple ratio (1) was computed for all VN words. The log ratio was used to give a linear scale. If verb usage outnumbers noun usage to a certain extent, i.e., when $r >> 0$, it suggests that the word is originally a verb and has just started to shift. If verb usage and noun usage are more or less equal, i.e., when $r \approx 0$, then either the shift is mature enough or there is genuine ambiguity. If noun usage outnumbers verb usage by a lot, i.e., when $r << 0$, it would mean that either the verb has over shifted or the word is originally a noun and is occasionally denominalized.

$$r = \log_2 \frac{\text{verb uses}}{\text{noun uses}} \quad (1)$$

B. Results

The results from Hong Kong, Beijing and Taipei are shown in Table IX. The column “No. in VN Shift” indicates the amount of VN words out of all verbs. Out of these we analyzed those with total frequency (including both verb and noun usages) 5 or more for their r values, and the results are shown in the last three columns. Each place has about 60% of the words reaching this threshold. With $r \geq 1$, verb usage at least doubles noun usage. With $1 > r > -1$, verb and noun usages are quite balanced. With $r \leq -1$, noun usage at least doubles verb usage. The results thus suggest that in real use, there are about 3-4% more nominalized uses of verbs found in Beijing than in Hong Kong and Taipei, which indicate quite a different, if not innovative, style of writing in Beijing. The figures also reflect the asymmetry between deverbalization of verbs and denominalization of nouns.

TABLE IX. SUMMARY OF RESULTS (HONG KONG, BEIJING & TAIPEI)

Source	No. of VN	$r \geq 1$	$1 > r > -1$	$r \leq -1$
		(Only for word types with freq ≥ 5)		
Hong Kong	14.4%	51.6%	26.0%	22.4%
Beijing	18.5%	45.5%	29.7%	24.8%
Taipei	15.5%	49.1%	27.1%	23.8%

The general observation is that Beijing demonstrates more nominalized usages of verbs than the other two communities. In addition, for Hong Kong and Taipei, on average about 50% of the VN words are just beginning to shift (with $r \geq 1$) and their verb usages are still dominant. On the contrary, only about 35% on average of the VN words

have $r \geq 1$ for Beijing. In other words, many words which are verbs originally do not actually play the role of verbs in Beijing. This might suggest that Chinese grammar is more seriously Europeanized in the Mainland.

We also found that 105 VN words are shared by all three communities and their usages are quite different among the three communities. Some examples and the corresponding r values are shown in Table X.

TABLE X. EXAMPLES OF VN WORDS COMMON TO THREE PLACES

VN Word	Hong Kong	Beijing	Taipei
發揮 (to express)	0.8074	2.7004	3.3219
經營 (to run a business)	1.5850	-0.1375	1.8074
宣傳 (to promote)	-1.3785	1.5850	0.4854
合作 (to co-operate)	1.3785	-1.1635	0.4594
衝擊 (to attack)	-0.3219	1.3219	-3.3219
感受 (to experience)	2.8074	-1.4150	-1.3219

Table X shows that *發揮* (*fa-hui*, to express) maintains most of its verb usage in Beijing and Taipei, but is considerably balanced with its noun usage in Hong Kong. On the other hand, *感受* (*gan-shou*, to experience) is mostly used as a verb in Hong Kong, but its noun usage predominates in Beijing and Taipei.

The above comparison indicates that the categorial fluidity phenomenon is relatively most common in Beijing – up to more than 18% of verbs undergo the verb-noun transitional process to various extents. These findings not only improve our understanding of this perennial problem in contemporary Chinese, but also have important implications for meaningful natural language applications.

VII. SYNTACTIC CHANGE

Besides lexical development, LIVAC also allows us to monitor syntactic change of the Chinese language. With its synchronic nature, we can trace how the new syntactic feature originates. In the following, we discuss the transitive verb 打造 (*da-zao*, to fabricate).

In the *Dictionary on Modern Chinese* published in 2003, the verb 打造 has the following definition:

“to fabricate (mostly metallic objects such as tools and ships)”

In a later edition published in 2005, one more definition has been added: “to create or to accomplish something such as brand names or company image”.

These two definitions show that the objects of 打造 change from *concrete* to *less concrete* or even *abstract*. It is thus meaningful to trace how this property of the syntactic argument changed. To be more specific, can we trace which Chinese community instigates this change first?

In LIVAC, the objects of 打造 are classified into 3 types:

- (a) Concrete objects, such as ships, furniture
- (b) Semi-concrete objects such as the aircraft carrier of the automotive business.
- (c) Abstract objects, such as New Taiwan, Peking operas, new brand names, new vista, new life, etc.

When we compare the change of the syntactic object in terms of abstractness for 打造 across the three communities,

we find that Taipei was the first to take on *abstract* objects for this verb, followed by Hong Kong and Singapore, then Shanghai and Beijing (non-government publications). The chronological order is summarized in Figure 2.



Figure 2. Chronological development of the abstractness of the syntactic object for 打造 (to fabricate) across Chinese communities²

VIII. CONCLUDING REMARKS

In this paper, we have drawn attention to the innovative use of a gigantic corpus of Chinese which has been cultivated synchronously and homothematically and with the introduction of a Windows approach. Such a linguistic corpus provides much more value than for traditional language modeling efforts in NLP applications such as IR and named-entity recognition. It can function usefully for data mining as well as monitoring linguistic variations in spatial and temporal dimensions which is uncommon for traditionally morphology rich languages, as well as to monitor the deeper concomitant developments in the larger relevant social and cultural contexts with the associated language users. It is hoped that with the addition of more mature applications of data mining techniques, much more findings can be reported in future.

ACKNOWLEDGMENT

We have benefited from input by many colleagues: W. F. Tsoi, K. P. Chow, Tom Lai, Terence Chan, Amy Liu, and colleagues from the ChiLin Star Corporation in Zhuhai, PRC.

REFERENCES

- [1] K. Church and R. Mercer, "Introduction to the Special Issue on Computational Linguistics Using Large Corpora," *Computational Linguistics*, vol. 19.1, 1993, pp. 1-24.
- [2] T. McEnery and A. Wilson, *Corpus Linguistics*. Edinburgh: Edinburgh University Press, 1996.
- [3] D. Biber, S. Conrad, and R. Reppen, *Corpus Linguistics: Investigating Language Structure and Use*. Cambridge: Cambridge University Press, 1998.
- [4] <http://www.livac.org>
- [5] B. Tsou, H. L. Lin, T. Chan, J. Hu, C. H. Chew, and J. K. P. Tse, "A Synchronous Chinese Language Corpus from Different Speech Communities: Construction and Application," *International Journal of Computational Linguistics and Chinese Language Processing*, vol. 2.1, 1997, pp. 91-104.
- [6] B. Tsou, *A Synchronous Dictionary on Pan-Chinese Syntactic Information*, unpublished [in Chinese].
- [7] B. Tsou and T. B. Y. Lai, "Chinese synchronous corpus and information mining," in *Critical Issues in Chinese Information Processing*, B. Xu, M. Sun and G. Jin, Eds. Beijing: Science Press, 2003, pp. 147-165 [in Chinese].
- [8] B. Tsou, "A window on re-lexification in Chinese," in *In Memory of Professor Li Fang-kuei: Essays on Linguistic Change and the Chinese Dialects*, P. H. Ting and A. Yue, Eds. Seattle/Taipei: University of Washington/Academia Sinica, 2000, pp. 53-72.
- [9] T. A. van Dijk, *News Analysis: Case Studies of International and National News in the Press*. Hillsdale: Lawrence Erlbaum, 1998.
- [10] T. A. van Dijk, *News as Discourse*. Hillsdale: Lawrence Erlbaum, 1998.
- [11] P. Garrett and A. Bell, "Media discourse: A critical overview," in *Approaches to Media Discourse*, A. Bell and P. Garrett, Eds. Oxford: Blackwell, 1998, pp. 1-20.
- [12] R. Fowler, *Language in the News: Discourse and Ideology in the Press*. London: Routledge, 1991.
- [13] B. Tsou and L. Feng, "A comparative study on neologisms in Chinese and Japanese: Towards a windows approach on the creation of neologisms and relexification," *Studies on Language*, vol. 3, 2000, pp. 51-70 [in Chinese].
- [14] B. Tsou and A. Chin, "A large synchronous corpus as monitoring corpus: Some comparative content analysis of Chinese and Japanese language developments," in *Proceedings of the 4th International Universal Communication Symposium (IUCS 2010)*, IEEE Computer Society, in press.
- [15] M. Sun, H. Huang, and J. Fang, "Identifying Chinese names in unrestricted texts," *Journal of Chinese Information Processing*, vol. 9.2, 1998, pp. 16-27.
- [16] L. Cheung and B. Tsou, "Personal names in unrestricted Chinese texts: nature and identification," in *Proceedings of Workshop on International Standards of Terminology and Language Resource Management, Third International Conference on Language Resources and Evaluation*, Las Palmas, Spain, May 28 – June 2, 2002, pp. 1-7.
- [17] B. Tsou and O. Kwong, "Aspects of MT requirements related to proper nouns in the Asian context" in *Proceedings of Workshop on Survey on Research and Development of Machine Translation in Asia*, 2002, pp. 85-93.
- [18] R. Song and B. Tsou, "A preliminary study on Chinese proper names," in *Proceedings of the 20th Anniversary Conference of CIPSC*, 2000, pp. 14-19. [in Chinese].
- [19] B. Tsou and R. J. You, *A Dictionary of Chinese New Words in the 21st Century*, Shanghai: Fudan University Press, 2007. [in Chinese]
- [20] B. Tsou and R. J. You, *A Dictionary of Chinese New Words*, Beijing: Commercial Press, 2010. [in Chinese]
- [21] F. Masini, *The Formation of Modern Chinese Lexicon and its Evolution toward a National Language: The Period from 1840 to 1898*. *Journal of Chinese Linguistics*, Monograph 6, Berkeley: *Journal of Chinese Linguistics*, 1993.
- [22] L. Wang, *A Sociolinguistic Study on Chinese Words*, Beijing: Commercial Press, 2003. [in Chinese].
- [23] W. Pan, B. Ye, and Y. Han, *A Study on Word Formation in Chinese*, Taipei: Students Publisher, 1993. [in Chinese].
- [24] J. Zhou, *The Meaning and Structure of Words*, Tianjin: Tianjin guji chubanshe. 1994. [in Chinese].
- [25] S. Lü, "A preliminary study on disyllabicity in modern Chinese," in *Collection of Essays on Chinese Grammar*, Beijing: Commercial Press, 1999, pp. 415-444. [in Chinese].
- [26] Z. Tang, *The Lexicon of Contemporary Chinese: Its Synchronic Situation and Change*, Shanghai: Fudan University Press, 2001. [in Chinese].
- [27] O. Kwong and B. Tsou, "A synchronous corpus-based study of verb-noun fluidity in Chinese," in *Proceedings of the 17th Pacific Asia Conference*, October 2003, pp. 194-203.
- [28] B. Tsou, A. Chin, and O. Kwong, "On incipient linguistic variations in chinese: A corpus approach," unpublished.

² A more detailed study on monitoring syntactic change in the Chinese language with the LIVAC corpus can be found in [28].

Efficient Access to Non-Sequential Elements of a Search Tree

Lubomir Stanchev

Computer Science Department

Indiana University - Purdue University Fort Wayne

Fort Wayne, IN, USA

stanchel@ipfw.edu

Abstract—This article describes how a search tree can be extended in order to allow efficient access to predefined subsets of the stored elements. This is achieved by marking some of the elements of the search tree with marker bits. We show that our approach does not affect the asymptotic logarithmic complexity for existing operations. At the same time, it is beneficial because the modified search tree can now efficiently support requests on predefined subsets of the search elements that it previously could not.

Keywords-marker bits; search trees; data structures

I. INTRODUCTION

A balanced search trees, such as an AVL tree ([1]), an AA tree (see [2]), or a B^+ tree ([3]), allows efficient retrieval of elements that are consecutive relative to an in-order traversal of the tree. However, there is no obvious way to efficiently retrieve the elements that belong to a predefined subset of the stored elements if they are not sequential in the search tree. For example, consider a database that stores information about company employees. A search tree may store information about the employees ordered by age. This search tree can be used to retrieve all the employees sorted by age, but the search tree does not efficiently support the request of retrieving all rich employees (e.g., making more than 100,000 per year) sorted by age. In this paper, we will show how the example search tree can be extended with marker bits so that both requests can be efficiently supported.

The technique that is proposed in this paper will increase the set of requests that can be efficiently supported by a search tree. This means that fewer search trees will need to be built. This approach will not only save space, but will also improve update performance.

Naïve solutions to the problem fail. For example, it is not enough to mark all the nodes of the search tree that contain data elements that belong to subsets of the data that we are interested in. This approach will not allow us to prune out any subtrees because it can be the case that the parent node does not belong to an interesting subset, but the child nodes do.

To the best of our knowledge, detailed explanation of how marker bits work have not been previously published. Our previous work [5] briefly introduces the concept of marker bits, but it does explain how marker bits can be maintained after insertion, deletion and update. Other existing

approaches handle requests on different subsets of the search tree elements by exhaustive search or by creating additional search trees. However, the second approach leads to not only unnecessary duplication of data, but also slower updates to multiple copies of the same data.

Given a subset of the search elements S , our approach marks every node in the tree that contains an element of S or that has a descendant that contains an element of S . These additional marker bits will only slightly increase the size of the search tree (with one bit per tree node), but will allow efficient logarithmic execution of requests that ask for the elements of S in the tree order.

In what follows, Section II presents core definitions, Section III describes how to perform different operations on a search tree with marker bits, and Section IV contains the conclusion.

II. DEFINITIONS

Definition 1 (MB-tree): An *MB-tree* has the following syntax: $\langle\langle S_1, \dots, S_s \rangle, S, O \rangle$, where S and $\{S_i\}_{i=1}^s$ are sets over the same domain Δ , $S_i \subseteq S$ for $i \in [1..s]$, and O is a total order over Δ . This represents a balanced search tree of the elements of S (every node of the tree stores a single element of S), where the in-order traversal of the tree produces the elements according to the order O . In addition, every node of the tree contains s marker bits and the i^{th} marker bit is set exactly when the node or one of its descendants stores an element that belongs to S_i - we will refer to this property as the *marker bit property*.

The above definition can be trivially extended to allow an MB-tree to have multiple data values in a node, as is the case for a *B Tree*, but this is beyond the scope of this paper.

Going back to our motivating example, consider the MB-tree $\langle\langle RICH_EMPS \rangle, EMPS, \langle age \rangle \rangle$. This represents a search tree of the employees, where the ordering is relative to the attribute *age* in ascending order. The *RICH_EMPS* set consists of the employees that make more than \$100,000 per year. Figure 1 shows an example instance of this MB-tree. Each node of the tree contains the name of the employee followed by their age and salary.

Each node in the MB-tree contains the name of the employee, their age, and their salary. Above each node the value of the marker bit is denoted, where the bit is set

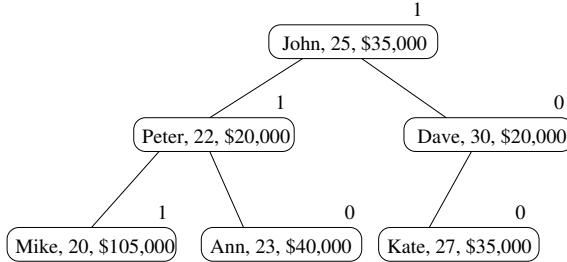


Figure 1. Example of an MB-tree

(operation)	(return value)
left()	left child
right()	right child
parent()	parent node
data()	stored data
m[i]	the i marker bit ($1 \leq i \leq s$)

Table I
INTERFACE OF A NODE

exactly when the node or one of its descendants contains a rich employee. As the figure suggests, the subtree with root node that contains the name Dave can be pruned out when searching for rich employees because the marker bit of the root node is not set. We will show that this MB-tree can be used to efficiently find not only all employees sorted by age, but also all rich employees sorted by age.

III. OPERATIONS ON AN MB-TREE

Although an MB-tree does not need to be binary, in the following discussion we will consider only binary trees for simplicity. In particular, we will assume that every node of the search tree supports the methods of the interface shown in Table I in constant time, where $\{S_i\}_{i=1}^s$ are the marker bit sets.

Next, we describe how the algorithms for tree search and update can be extended in the presence of marker bits.

A. Element Insertion

After an algorithm has inserted a leaf node n , it should call the `insert_fix` method from Algorithm 1 to update the marker bits in the tree.

Algorithm 1 `insert_fix` (Node n)

```

1: for  $i \leftarrow 1$  to  $s$  do
2:   if  $n.\text{data}() \in S_i$  then
3:      $n.m[i] \leftarrow 1$ 
4:   else
5:      $n.m[i] \leftarrow 0$ 
6:   end if
7: end for
8: insert_parent_fix( $n.\text{parent}()$ ,  $n.m$ )
  
```

Lines 1-7 of the code set the marker bits for the new node. The call to the recursive function `insert_parent_fix` fixes the marker bits of the ancestors of the inserted node, where the later is presented in Algorithm 2.

Algorithm 2 `insert_parent_fix` (Node n , Bit[] m)

```

1: if  $n = \text{null}$  then
2:   return
3: end if
4: changed  $\leftarrow \text{false}$ 
5: for  $i \leftarrow 1$  to  $s$  do
6:   if  $m[i] = 1$  and  $n.m[i] = 0$  then
7:      $n.m[i] \leftarrow 1$ 
8:     changed  $\leftarrow \text{true}$ 
9:   end if
10: end for
11: if changed then
12:   insert_parent_fix( $n.\text{parent}()$ ,  $n.m$ )
13: end if
  
```

We claim that the resulting tree satisfies the marker bit property. In particular, note that only the marker bits of the inserted node and its ancestors can be potentially affected by the insertion. Lines 1-7 of the `insert_fix` method update the marker bits of the node that is inserted. If the i^{th} marker bit of the node is set, then we check the i^{th} marker bit of its parent node (Lines 6 of the `insert_parent_fix` method). If the i^{th} marker bit of the parent is set, then the i^{th} marker bit of all ancestors will be set because of the marker bit property and nothing more needs to be done for the i^{th} marker bit. Conversely, if the i^{th} marker bit of the parent is not set, then we need to set it and then check the i^{th} marker bit of the parent of the parent node. This is done by Line 7 and the recursive call at Line 12, respectively. The variable `changed` is used to record whether any of the marker bits of the current node have been changed. If the variable is not changed, then the marker bits of the ancestor nodes will not need to be updated. Therefore, the marker bits of the inserted node and its ancestors are updated correctly and the marker bit property holds for the updated search tree.

B. Deleting a Node with Less than Two Children

Deleting a node with two children from a binary tree cannot be performed by just connecting the parent of the deleted node to the children of the deleted node because the parent node may end up with three children. Therefore, we will consider two cases: when the deleted node has less than two non-null children and when the deleted node has two non-null children. The first case is explained next, while the second case is explained in Section III-D.

An implementation of Algorithm 3 should be called before a node n with less than two non-null children is deleted. In the algorithm, $n.\text{child}()$ is used to denote the

non-null child of n and $m[i]$ is set when the i^{th} marker bit of the ancestor nodes need to be checked. The algorithm for the method `delete_parent_fix` that updates the marker bits of n 's ancestors in the search tree is shown in Algorithm 4.

Algorithm 3 `delete_fix_simple`(Node n)

```

1: for  $i \leftarrow 1$  to  $s$  do
2:   if  $n.\text{data}() \in S_i$  and ( $n$  is leaf node or
    $n.\text{child}().m[i] = 0$ ) then
3:      $m[i] \leftarrow 1$ 
4:   else
5:      $m[i] \leftarrow 0$ 
6:   end if
7: end for
8: delete_parent_fix( $n.\text{parent}()$ ,  $m$ )

```

Algorithm 4 `delete_parent_fix`(Node n , Bit[] m)

```

1: if  $n = \text{null}$  then
2:   return
3: end if
4:  $\text{changed} \leftarrow \text{false}$ 
5: for  $i \leftarrow 1$  to  $s$  do
6:   if  $m[i] = 1$  and  $n.\text{data}() \notin S_i$  and ( $n$  has no other
   child or  $n.\text{other\_child}().m[i] = 0$ ) then
7:      $n.m[i] \leftarrow 0$ 
8:      $\text{changed} \leftarrow \text{true};$ 
9:   end if
10: end for
11: if  $\text{changed}$  then
12:   delete_parent_fix( $n.\text{parent}()$ ,  $m$ )
13: end if

```

Note that we have used $n.\text{other_child}$ to denote the child node of n that is not on the path to the deleted node. We claim that the deletion algorithm preserves the marker bit property. In particular, note that only the ancestors of the deleted node can be affected. If $m[i] = 1$ (Line 6 of the `delete_parent_fix` method), then we check whether the data in the node belongs to S_i and whether the i^{th} marker bit of the other child node is set. If both conditions are false, then the only reason the i^{th} marker bit of n is set is because the data in the deleted node belonged to S_i and now this marker bit needs to be unset (Line 7) and the ancestors of n needs to be recursively checked (Line 12). Conversely, if one of the conditions is true or $m[i] = 0$, then the i^{th} marker bit of n and its ancestors will not be affected by the node deletion. Therefore, the marker bits of the ancestors of the deleted node are updated correctly and the marker bit property holds for the updated search tree.

C. Element Update

Algorithm 5 should be executed after the data in a node n is modified, where v is the old data value of n .

Algorithm 5 `update_fix`(Node n , Value v)

```

1:  $old \leftarrow n$ 
2: for  $i = 1$  to  $s$  do
3:   if  $n.\text{data}() \in S_i$  or ( $n.\text{left}() \neq \text{null}$  and
    $n.\text{left}().m[i] = 1$ ) or ( $n.\text{right}() \neq \text{null}$  and
    $n.\text{right}().m[i] = 1$ ) then
4:      $n.m[i] \leftarrow 1$ 
5:   else
6:      $n.m[i] = 0$ 
7:   end if
8:   if  $n.m[i] = 1$  and  $old.m[i] = 0$  then
9:      $m[i] \leftarrow \text{"insert"}$ 
10:    else if  $n.m[i] = 0$  and  $old.m[i] = 1$  then
11:       $m[i] \leftarrow \text{"delete"}$ 
12:    else
13:       $m[i] \leftarrow \text{"no change"}$ 
14:    end if
15: end for
16: update_parent_fix( $n.\text{parent}()$ ,  $m$ )

```

The pseudo-code updates the marker bits of the node n and then calls the `update_parent_fix` method, which is presented in Algorithm 6.

Algorithm 6 `update_parent_fix`(Node n , Value[] m)

```

1: if  $n = \text{null}$  then
2:   return
3: end if
4:  $\text{changed} \leftarrow \text{false}$ 
5: for  $i = 1$  to  $s$  do
6:   if  $m[i] = \text{"insert"}$  and  $n.m[i] = 0$  then
7:      $n.m[i] \leftarrow 1$ 
8:      $\text{changed} \leftarrow \text{true}$ 
9:   end if
10:  if  $m[i] = \text{"delete"}$  and  $n.\text{data}() \notin S_i$ 
   and ( $n.\text{other\_child}() = \text{null}$  or
    $n.\text{other\_child}().m[i] = 0$ ) then
11:     $n.m[i] \leftarrow 0$ 
12:     $\text{changed} \leftarrow \text{true}$ 
13:  end if
14: end for
15: if  $\text{changed}$  then
16:   update_parent_fix( $n.\text{parent}()$ ,  $m$ )
17: end if

```

Note that we have used $n.\text{other_child}()$ to denote the child node of n that is not on the path to the updated node. The method `update_fix` preserves the marker bit

property because it is a combination of the `insert_fix` and `delete_fix_simple` methods. In particular, $m[i]$ in the method `update_fix` is set to `insert` when the i^{th} marker bit of the updated node was changed from 0 to 1 and to `delete` when this marker bit was updated from 1 to 0. The first case is equivalent to a node with the i^{th} marker bit set being inserted, while the second case is equivalent to a node with the i^{th} marker bit set being deleted.

D. Deleting a Node with Two Children

As it is usually the case ([4]), we assume that the deletion of a node n_1 with two non-null children is handled by first deleting the node after n_1 relative to the tree order, which we will denote as n_2 , followed by changing the data value of n_1 to that of n_2 . The pseudo-code in Algorithm 7, which implementation should be called after a node is deleted from the tree, shows how the marking bits can be updated, where initially $n = n_1$, p is the parent of n_2 , v is the value of the data that was stored in n_2 , and $m[i] = 1$ exactly when $n_2.m[i] = 1$ and for all descendants of n_2 , $m[i] = 0$.

Algorithm 7 `delete_fix_two_children(n, p, v, m)`

```

1: if  $p = n$  then
2:   update_fix( $n, v$ )
3: end if
4:  $changed \leftarrow \text{false}$ 
5: for  $i=1$  to  $s$  do
6:   if  $m[i] = 1$  and  $p.\text{data} \notin S_i$  and ( $p$  has no other
      child or  $p.\text{other\_child}().m[i] = 0$ ) then
7:      $p.m[i] \leftarrow 0$ 
8:      $changed \leftarrow \text{true}$ 
9:   end if
10: end for
11: if  $changed$  then
12:   delete_fix_two_children( $n, p.\text{parent}(), v, m$ )
13: else
14:   update_fix( $n, v$ )
15: end if

```

In the above code “ p has no other child” refers to the condition that p has no other child than the child that it is on the path to the deleted node n_2 . Similarly, $p.\text{other_child}()$ is used to denote the child of p that is not on the path to the deleted node n_2 . Note that the above algorithm changes the nodes on the path from n_2 to n_1 using the deletion algorithm from method `delete_parent_fix` and the nodes on the path from n_1 to the root of the tree using the update algorithm from the method `update_fix` and is therefore correct.

E. Tree Rotation

Most balancing algorithms (e.g., the ones for AVL, red-black, or AA trees) perform a sequence of left and/or right

rotations whenever the tree is not balanced as the result of some operation. Here, we will describe how a right rotation can be performed, where the code for a left rotation is symmetric. The pseudo-code in Algorithm 8 should be called with a parent node n_2 and right child node n_1 after the rotation around the two nodes was performed.

Algorithm 8 `rotate_right_fix(n1, n2)`

```

1: for  $i \leftarrow 1$  to  $s$  do
2:   if  $n_1.\text{data}() \in S_i$  or ( $n_1$  has left child and
       $n_1.\text{left}().m[i] = 1$ ) then
3:      $n_1.m[i] \leftarrow 1$ 
4:   end if
5:   if  $n_2.\text{data}() \in S_i$  or ( $n_2$  has left child and
       $n_2.\text{left}().m[i] = 1$ ) or ( $n_2$  has right child and
       $n_2.\text{right}().m[i] = 1$ ) then
6:      $n_2.m[i] \leftarrow 1$ 
7:   end if
8: end for

```

The above pseudo-code only fixes the marker bits of n_1 and n_2 . The descendants of all other nodes will not change and therefore their marker bits do not need to be updated.

F. Time Analysis for the Modification Methods

Obviously, the pseudo-code for the rotation takes constant time. The other methods for updating marker bits visit the node and possibly some of its ancestors and perform constant number of work on each node and therefore take order logarithmic time relative to the number of nodes in the tree. Therefore, the extra overhead of maintaining the marker bits will not change the asymptotic complexity of the modification operations.

G. Search

Let us go back to our motivating example from Figure 1. Our desire is to efficiently retrieve all rich employees in the tree order. This can be done by repeatedly calling the implementation of the `next` method from Algorithm 9. The terminating condition is when the method returns `null`. The algorithm finds the first node that is n or that is after n , relative to the tree order, and that has data that belongs to the set S_i , where d is initially set to `false`.

The algorithm first checks if the data in the current node is in S_i . If it is, then we have found the resulting node and we just need to return it. Next, we check the left child node. If we did not just visit it and its i^{th} bit is marked and it is after the start node relative to the in-order tree traversal order, then the subtree with root this node will contain a node with data in S_i that will be the resulting node. Next, we check if the right child has its i^{th} bit marked. This condition and the condition that we have not visited it before guarantees that this subtree will contain the resulting node. Finally, if neither of the child subtrees contain the node we

Algorithm 9 next (n, i, d)

```

1: if ( $n.\text{data}() \in S_i$ ) then
2:   return  $n$ 
3: end if
4: if  $n.\text{left}()$  is not the last node visited and
    $n.\text{left}() \neq \text{null}$  and  $n.\text{left}().m[i] = 1$  and  $d$ 
   then
5:   return next ( $n.\text{left}(), i, \text{true}$ )
6: end if
7: if  $n.\text{right}()$  is not the last node visited and
    $n.\text{right}() \neq \text{null}$  and  $n.\text{right}().m[i] = 1$  then
8:   return next ( $n.\text{right}(), i, \text{true}$ )
9: end if
10: if  $n.\text{parent}() = \text{null}$  then
11:   return null
12: end if
13: return next ( $n.\text{parent}(), i, d$ )

```

are looking for, we start checking the ancestor nodes in order until we find an ancestor that has a right child node that we have not visited and its i^{th} marker bit for this child is set. We then visit this subtree because we are guaranteed that it will contain the resulting node. Therefore, the algorithm finds the first node starting with n that has data is in S_i . Since, in the worst case, we go up a path in the search tree and then down a path in the search tree, our worst-case asymptotic complexity for finding the next node with data in S_i is logarithmic relative to the size of the tree, which is the same as the asymptotic complexity of the traditional method for finding a next element in a search tree.

IV. CONCLUSION

We introduced MB-trees and showed how they are beneficial for accessing predefine subsets of the tree elements. MB-trees use marker bits, which add only light overhead to the different operations and do not change the asymptotic complexity of the operations. An obvious application of MB-trees is merging search trees by removing redundant data, which can result in faster updates because fewer copies of the redundant data need to be updated.

REFERENCES

- [1] G. M. Adelson-Velskii and E. M. Landis, “An Algorithm for the Organization of Information,” *Soviet Math. Doklady*, vol. 3, pp. 1259–1263, 1962.
- [2] A. Andersson, “Balanced search trees made simple,” *Workshop on Algorithms and Data Structures*, pp. 60–71, 1993.
- [3] R. Bayer and E. McCreight, “Organization and Maintenance of Large Ordered Indexes,” *Acta Informatica*, vol. 1, no. 3, 1972.
- [4] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*. McGraw Hill, 2002.
- [5] L. Stanchev and G. Weddell, “Saving Space and Time Using Index Merging,” *Elsevier Data & Knowledge Engineering*, vol. 69, no. 10, pp. 1062–1080, 2010.

An Optimistic Transaction Model for a Disconnected Integration Architecture

Tim Lessner*, Fritz Laux†, Thomas Connolly*, Cherif Branki*, Malcolm Crowe*, and Martti Laiho‡

*School of Computing, University of the West of Scotland, name.surname@uws.ac.uk

†Fakultät Informatik, Reutlingen University, Germany, fritz.laux@reutlingen-university.de

‡ Dpt. of Business Information Technology, Haaga-Helia University of Applied Sciences, Finland, martti.laiho@haaga-helia.fi

Abstract—This work presents a disconnected transaction model able to cope with the increased complexity of long-living, hierarchically structured, and disconnected transactions. We combine an Open and Closed Nested Transaction Model with Optimistic Concurrency Control and interrelate flat transactions with the aforementioned complex nature. Despite temporary inconsistencies during a transaction's execution our model ensures consistency.

Index Terms—Disconnected Transaction Management; Optimistic Concurrency Control; Advanced Transaction Models

I. INTRODUCTION

Nowadays, Transaction Management (TM) must not only deal with short lived and flat transactions, TM must provide a transactional execution of long running and hierarchically structured business processes, so called complex transactions, involving many distributed loosely coupled, heterogeneous, and autonomous systems, represented as services as in Service Oriented Computing (SOC) for instance. To facilitate the loose coupling and increase the autonomy data should be modified in a disconnected and not in an online manner. By this we mean that the set of proposed modifications to data is prepared offline that requires a transacted sequence of operations on separate database connections. Further, message exchange, which takes place in such an architecture, is asynchronous and so is the data access, too. From a transactional view, the execution of a business process is a tree of interdependent and interleaved transactions, and during its execution the ACID (Atomicity, Consistency, Isolation, Durability) [1] properties are often weakened, and they allow for temporary inconsistencies to increase the performance. A locking isolation protocol, for instance, where other concurrent transactions read only committed results, leads to long blocking time caused by the process's duration and the asynchronous message exchange typical for disconnected architectures.

A disconnected nature overcomes this challenge for the price of a weakened isolation. The drawback of a weakened isolation is that other transactions can read pending results, which increases the danger for data to become inconsistent, and a widely used solution is compensation to semantically undo effects, also cascading effects, even after the transaction technically commits. Compensation can be interpreted as a necessity to conform to reality.

There are a couple of scenarios where a transaction becomes complex. The booking of a journey, for example, involves several heterogeneous web applications. Also cloud computing, where flat transactions access a highly replicated and distributed data storage, is confronted with a complex transaction structure.

Before the paper introduces the Transaction Model in section III, the problem is described by the next section, as well as the contribution of the paper. Related work is discussed in section IV and the Conclusion section completes this paper.

II. PROBLEM DESCRIPTION & CONTRIBUTION

One challenge is to provide a consistent transactional integration of heterogeneous systems, applications and databases across database (DB), middleware (MW) and application layer.

A lot of work has been carried out regarding (i) the transactional integration of heterogeneous systems (applications or databases) by the research community and industry (see [2][3][4]). In the relatively new domain of SOC many Web Service specifications cover a plurality of aspects and they all have in common that they provide a transactional interaction to a certain degree between services, or more generally components. These models have been motivated by the need for Business (BT) or User Transactions (UT). Moreover, (ii) transaction processing in DBMS has been addressed by a vast amount of research which focuses on the correctness, hence the serializability, of concurrent data processing in a transparent way by considering write and read operations only.

However, we believe the problem is that a common formal model integrating both aspects (i,ii) is missing, especially a model that considers the flat transactions as implemented by components and the overlying complex structure of transactions which guarantees consistency even if isolation or atomicity is weakened.

The contribution of this paper is a formal transaction model that interrelates the flat transactions as implemented by components with their composition and the resulting complex transaction structure. We impose an Optimistic Concurrency Control (OCC) structure within a component and require an OCC at the MW layer. The approach allows the detection of inconsistencies and furthermore decouples the concurrency control (CC) mechanism of the DB layer and thereby provides a solution for the “Impedance Mismatch” between (i) and

(ii). We also believe that such a more integrated model helps to determine a trade-off between consistency, availability, and failure tolerance –user expectations– of a transactional integration system.

III. TRANSACTION MODEL

We define a transaction t as a composition of several flat and short living transactions t_1, t_2, \dots, t_n where components $COMP = comp_1, \dots, comp_j$ implement these transactions t_n . As in some MW specifications like the Java Enterprise Edition (JEE), components become part of the transactions, and they are in a transaction scope and bound to the life cycle of the component itself. In a disconnected architecture, however, these components lose their context in terms of the transaction as soon as the data is delivered. Components communicate with a data access layer asynchronously and no locks are kept because of the long-living and disconnected nature.

The overview in figure 2 shows the dependencies between the different concepts defined in this section.

A. Disconnected Transaction

A disconnected transaction has a read sphere sph^r and a write sphere sph^w . A sphere $sph \in SPH$ is defined as a set $sph = \{t_1, \dots, t_n\}$ of transactions which logically groups transactions that belong together. Here, sph is a set of flat, short living, ACID transactions, and let transaction t be a sequence of operations $t = (op_1, \dots, op_m)$ with $OP = \{op_1, \dots, op_m\}$ as the set of data operations where each op_m is either of type $read(DO_m)$ or $write(DO_m)$, and let DO be the set of all data objects $DO = \cup_{k \in K} DO_k, K = \{1, 2, \dots, m\}$. If there are versions of DO_k we denote a version DO_k^v , with superscript.

The disconnected behaviour is defined by $sph^r \cap sph^w = \emptyset$ and some transaction t is either in sph^r or sph^w .

Regarding the definitions introduced so far from an implementation point of view an implementation of dt , hence the several flat transactions in dt 's write and read sphere, is required to initiate read and write transactions. Therefore, we define $comp$ as a component that implements dt . Additionally, we define a component to be in one of the following phases: reading (p1), disconnected and working (p2), validating (p3), and writing (p4). Hence we define a component to be in the phases similar to the phases of OCC. The difference, however, is the explicit disconnected and working phase. Notice, we impose this structure on a $comp$ which may be seen as a design guideline for the implementation of a single component.

Reading can be described as loading all the data required. After the modifications take place validation starts. Validation must be interpreted as a pre-phase of writing and only transactions $T^w \subseteq sph^w$ are allowed to enter p3 and p4 (transactions $T^r \subseteq sph^r$ can only enter the read phase). By following this structure a $comp$ can be seen as a component that follows the OCC paradigm introduced by [5].

Within an implementation of $comp$, write transactions may depend on each other and an explicit execution order like $t_1 \rightarrow$

$t_2 \rightarrow t_4$ can be given. We define read transactions to not depend on each other because they can be parallelised without conflicting with each other.

Therefore, we define an explicit ordering over all writing transactions T^w and let Gdt be a directed, acyclic graph $Gdt = (Vdt, Edt)$ as defined in definition 1. The graph defines an execution order between two transactions $t_n \rightarrow t_o$ and Gdt is a partial order over the set of transactions $T'^w \subseteq (T^w \in sph^w)$. The set of transactions T''^w represents free transactions of a component $comp$, i.e., $T''^w \notin Vdt$. Hence, $T^w = T'^w \cup T''^w$.

The SAGA [6] model introduced the notion of compensation to semantically undo the effects of transactions. Compensation has been introduced to cope with the requirement for a weak isolation that arises if several sub-transactions form a long living process but each of the sub-transactions is allowed to commit and other transactions may read pending results. In case the transaction aborts its sub-transactions, whether committed or not, need to be undone, which is only possible by executing a compensation, e.g., to cancel a flight is the compensation of booking a flight. Our model also foresees compensation transactions to semantically undo the effects of a component, i.e., dt see III-C.

In our model a $comp$ either provides its own sequence of compensation transactions $comp_i^{-1} = (t_1, \dots, t_n)$ or points to another component $comp_i^{-1} = comp_j$ representing the compensation.

Finally, we define a disconnected transaction as:

Definition 1: Disconnected transaction dt

(1) A Disconnected transaction is defined as $dt := (sph^r, sph^w, Gdt, comp, comp^{-1})$ with read sphere $sph^r = (t_1, \dots, t_n)$, write sphere $sph^w = (t_1, \dots, t_m)$, a partial order $Gdt = (Vdt, Edt)$ with $Vdt \equiv T'^w \subseteq T^w$ and the set of edges Edt is defined as $\forall t_n, t_o \in V : t_n \rightarrow t_o \Leftrightarrow edt \in Edt$ with $edt = (t_n, t_o)$. Let $comp$ be the component that implements dt , and let $comp^{-1}$ be the compensation handler of dt . And, $\forall comp \in COMP : comp$ is in exactly one phase p1,p2,p3 or p4.

- (2) $\forall dt \in DT : sph^r \cap sph^w = \emptyset$
- (3) dt only commits successfully if all T^r and T^w commit successfully.
- (4) In the case of failure the $comp^{-1}$ must be executed to semantically undo the changes of dt .

B. Consistency of dt

Due to the long-living and disconnected nature it would be not favorable to lock data for as long as dt lasts because locking hinders global progress of the transaction. However, we must restrict this statement. No locking refers to the entire life-cycle (p1-p4) of dt , and locking for the short living transactions T is allowed, because they release their locks with their commit and so, the blocking time is reduced to the time validation and writing takes place (p3,p4). An exclusive access during the validation and writing phase is necessary because a consistent snapshot of a data object that has to be validated is required and modifications must be written back eventually.

We aim for a more decoupled transaction management. This in turn requires validation to prevent from read and write anomalies, like lost update. Our model foresees validation to take place at the MW layer. By applying validation at the MW layer we can basically decouple the DBMS in a sense that consistency is already provided by the MW. To ensure consistency at the MW the validation mechanism must perform a validation as introduced by Kung and Robinson[5].

Definition 2: Validation

(1) Let $RS(t_i)$ be the set of DO_i that is read by t_i of *comp* and let $WS(t_n)$ be the set of DO_m eventually modified by t_n of *comp*. Also, let $ts(RS(t_n))$ be the timestamp of the read set of t_n . Only if $ts(RS(t_n)) < ts(RS(t_o)) \wedge RS(t_n) \cap RS(t_o) = \emptyset \wedge WS(t_n) \cap WS(t_o) = \emptyset \wedge WS(t_n) \subseteq RS(t_n)$ holds, validation $val(DO_k^v) \rightarrow DO_k^{v+1}$ is conflict free. Let val be an algorithm that detects conflicts between two versions DO_k^v and DO_k^{v+1} of a data object by applying these aforementioned rules.

(2) Further we require the validation to be “escrow serializable” [7] and to obey the order defined by Gdt .

By applying this validation schema outdated data, i.e., data that has been modified by other transactions during the execution of dt , can be detected and anomalies can be prevented. In our previous work [7] about optimistic validation in disconnected and mobile computing we introduced “escrow serializability” ec and a “reconciliation mechanism” that allows the number of validation conflicts to be reduced by automatically replaying a certain class of operations. We require the validation to be ec , and the execution of each t is correct, if it is ec serializable.

Another issue which has to be considered is the possible existence of so called atomic units within sph^w . Atomic units exist if some transactions T^w are commit or abort dependent to each other, i.e., transaction t_n is only allowed to commit if transaction t_m does. To depict such dependencies we have to extend our model.

1) *Atomic Units*: An atomic unit groups several flat transactions where an atomic outcome of the group is required. The members of an atomic unit become sub-transactions. Considering the concept of atomicity the concept of an atomic unit is ambiguous because something that is atomic now consists of other atomic units, namely each sub-transaction itself. The point is that atomic refers to the expected outcome and the execution of several sub-transactions must be atomic. In other words, atomicity is relative to the level of the transaction tree.

We introduce the notion of an atomic unit $au = (t_1, \dots, t_n)$ as a sequence of flat transactions, and let $AU_i = \{au_{i,1}, \dots, au_{i,h}\}$ be the set of all atomic units of dt_i which form an imposed structure on sph_i^w ; thus $sph_i^w := AU_i$.

Definition 3: Atomic unit au of dt

(1) Let $AU_i = \{au_{i,1}, \dots, au_{i,h}\}$ be the set of all atomic units of dt_i which is an imposed structure on sph_i^w ; thus $sph_i^w := AU_i$. Further, let $au_{i,h} = (t_1, \dots, t_n)$ group $(t_1^w, \dots, t_n^w) \in T^w$ into an indivisible group of transactions

where either all $T^w \in au_{i,h}$ commit or abort. And, let AU be the set of all AU_i

Now, let $csDO_i$ be the change set –modifications– of data objects modified by dt_i and $csDO_i(au_{i,h})$ the change set of $au_{i,h}$. The validation must now ensure that only if each $csDO_i(au_{i,h})$ passes validation the modifications are written back to the database. To achieve an atomic outcome for sph_i^w we need a Closed Nested Transaction CNT [8], [9] structure that is able to guarantee an atomic outcome of sph_i^w . We need a CNT because of the imposed order and the atomic units which may encompass several t .

Definition 4: Closed Nested Transaction CNT

(1) Let $CNT(sph_i^w)$ be a closed nested transaction over sph_i^w which is defined as a tree $CNT(sph_i^w) = (Vcnt, Ecnt)$ with sph_i^w as root node r which only commits if all its children commit, the set of vertexes $Vcnt \equiv AU_i$ and the set of edges $Ecnt$ is defined as $\forall au_{i,h}, au_{i,j} \in Vcnt : au_{i,h} \rightarrow au_{i,j} \Leftrightarrow ecnt \in Ecnt$ with $ecnt = (au_{i,h}, au_{i,j})$.

(2) If there exists an order (edge) $\exists Edt \in Edt = t_m \rightarrow t_n$ with $t_m \in au_{i,h}$ and $t_n \in au_{i,j}$ for $h \neq j$ then there must be a dependency between $au_{i,h}$ and $au_{i,j}$ so that $t_m \rightarrow t_{first} \in au_{i,j}$, and $t_{last} \in au_{i,j} \rightarrow t_{m+1} \in au_{i,h}$, with t_{first} as the first and t_{last} as last element in $au_{i,j}$, and let t_{m+1} be the successor of t_m so that $t_m \rightarrow t_{m+1}$. Thus $au_{i,j}$ becomes part of $au_{i,h}$. If $\neg \exists (t_m \rightarrow t_{m+1})$ then $au_{i,j}$ runs concurrent to $au_{i,h}$.

(3) Further, the order of atomic units must terminate.

Example 1: CNT

Given $sph^w = (au_1, au_2, au_3, au_4, au_5)$ with $au_1 = (t_1, t_2, t_3, t_4)$, with $au_2 = (t_5, t_6)$, $au_3 = (t_7, t_8)$, $au_4 = (t_9)$, $au_5 = (t_{10})$, and

$$Gdt = (Vdt = \{t_1, t_3, t_4, t_5, t_6, t_7, t_8, t_9\}, Edt = \{(t_1, t_3)(t_3, t_6)(t_3, t_4)(t_5, t_6)(t_7, t_8)(t_7, t_9)\})$$

The resulting CNT , i.e., the execution model, is shown in Fig. 1. Notice, au 's are shown as dashed lines. As shown, au_2 becomes part of au_1 because of the edges (t_3, t_6) and (t_3, t_4) defined in Gdt . Since there is no order defined between t_1, t_2 it is possible to parallelise t_2 .

Regarding $au_3 = (t_7, t_8)$ and $au_4 = (t_9)$ the situation differs. Due to the order relations (t_7, t_8) and (t_7, t_9) au_3 and au_4 must be executed serial.

To achieve an atomic outcome of CNT a Two-Phase-Commit (2PC) is required between t_2 and the transactions t_3, t_5, t_6, t_4 with t_1 as coordinator. One possibility to achieve an atomic outcome of t_3, t_5, t_6, t_4 is to introduce a new atomic unit au'_1 . We regard this issue as implementation detail and leave an answer open for future research. However, a coordination between each chain in a branch is required. For example, the coordinator initiates t_3 , awaits the result, in case the result is a pre-commit, it initiates t_5 , then t_6 and finally t_4 if all of them sent their pre-commit au_2 is committed and the result is sent back as a pre-commit to t_1 . Between au_1 , au_3 , and au_5 a 2PC is required.

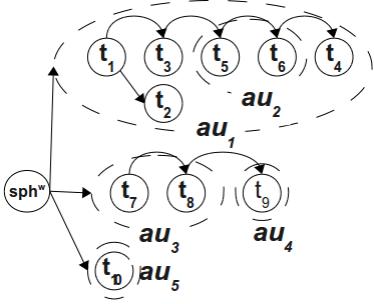


Fig. 1. Exemplary *CNT* (straight lines show possible parallelisation, curved lines show an order)

According to the example some further specification for *CNT* is required to ensure an atomic outcome:

Definition 5: Commitment rules of *CNT*

- (1) A parent is only allowed to commit if all its children commit.
- (2) Between all siblings of *CNT* a Two-Phase-Commit (2PC) is required.
- (3) Between a chain of *au* in a branch an external coordinator must ensure:
 - (a) A successor of $au_{i,h}$ is only allowed to start if its predecessor $au_{i,h-1}$ pre-commits.
 - (b) If one $au_{i,h}$ in the chain aborts all other *au* have to abort, too.
 - (c) If all *au* sent their pre-commit to the coordinator the coordinator sends a commit message to all *au* and each *au* has to commit, i.e., a commit is a promise by the *au*.
 - (d) A *DO* can be passed from a *au* only to its predecessor.

Notice the difference between a parallel execution which requires a 2PC and a chained one which requires coordination for an ordered execution. Both have in common that a commit is a promise that requires the pre-commit state and if only one of the involved parties aborts, the entire *au* must be aborted. Following the rules defined in Definition 5 compensation is not required, and recovery is possible because locks are released after a commit of all involved transactions (compensation will play an important role later in section III-C). Also, since a *DO* is only passed between already pre-committed, *au* isolation is maintained too. For further details in the domain of nested transactions we refer to the seminal work [8], [9], [10], [1], [11].

Now, we consider the concurrent execution of several *dt*. Usually a correct, i.e., consistent, concurrent, execution is given if the conflict graph between all transactions, *dt* in our case, is acyclic. Our model, however, does not require such a conflict graph and we rely on the optimistic validation instead.

The concurrent execution of several *au* of different *dt* is correct if OCC is able to detect conflicts between *au* that are either a sequence of transactions or single transactions, and if the order within and between atomic units is obeyed. Hence, an arbitrarily ordering of concurrent *au* is limited to the imposed order defined, but free transactions, i.e., atomic units, which

are not part of the order relation can interleave with ordered atomic units in any way. Further, bear in mind that OCC is a first wins strategy.

A concurrent execution of several $AU = (AU_i = (au_{i,1}, \dots, au_{i,j}), AU_k = (au_{k,1}, \dots, au_{k,l}), \dots)$ must be consistent because the OCC validation detects read-write or write-write conflicts for some data object *DO* (see Definition 2) and an arbitrary interleaving of free transactions with ordered *au*. In case of a conflict the transaction will be aborted. Further, the *CNT* ensures an atomic and non-isolated outcome. Hence, without providing a proof (see section V) OCC and *CNT* should ensure a consistent and atomic outcome of a concurrent execution of atomic units.

C. User Transaction *ut*

As already mentioned, a transactional integration has to deal with a complex hierarchically structured transaction, referred to as user transaction *ut*. An *ut* is a composition of several components that interact with each other in a defined order.

Definition 6: User transaction *ut*

Let $UT = ut_1, ut_2, \dots, ut_m$ be the set of all user transactions. We define an $ut_m := (DT_m, Gut_m, AUT_m)$ with DT_m as the set of disconnected transactions composed by *ut*. Gut_m is a directed, acyclic graph with $Gut_m = (Vut_m, Eut_m)$ with $Vut_m \equiv DT_m = dt_{1,m}, \dots, dt_{i,m}$ and the set of edges Eut_m is defined as $\forall dt_{m,i}, dt_{m,n} \in Vut_m : dt_{m,i} \rightarrow dt_{m,n} \Leftrightarrow eut \in Eut_m$ with $eut = (dt_{m,i}, dt_{m,n})$. The dependency $dt_{m,i} \rightarrow dt_{m,n}$ expresses an execution order, hence Gut_m represents a partial order over the set of disconnected transactions DT_m for ut_m . AUT is defined in the following section.

In the following we consider the existence of atomic units within an *ut* and how we can achieve an atomic outcome of *ut*.

D. Consistency of *ut*

As already investigated by the research community a complex transaction, like *ut*, must cope with sub-transactions defined by transactional boundaries within the execution model. A transactional boundary demarcates parts of the complete transaction and defines thereby sub-transactions, or in other words atomic units (atomic refers to the outcome). These units can now be interleaved in a concurrent execution which leads to an increased performance because not an entire *ut* must be scheduled.

Further, atomic units can be exploited to allow for a partial rollback of *ut* if the sequence of corresponding compensation steps is defined. This is also known as the “Spheres of Joint Compensation” model introduced by [12]. More details about the settings and the technological realisation of transactional boundaries can be found in [13], [14], [15].

We provide a more general notion of atomic units and focus on the interrelation between these more high level units and above defined disconnected transactions.

1) *Atomic Units of ut*: The following definitions are analog to the ones in section III-A above.

Definition 7: Atomic unit *aut* of *ut*

Let $AUT_m = \{aut_{m,1}, \dots, aut_{m,j}\}$ be the set of all atomic units of ut_m with $aut_{m,j} = (dt_1, \dots, dt_i)$, and with $(dt_1, \dots, dt_i) \in Eut_m$. Atomic unit $aut_{m,j}$ groups the spheres of disconnected transactions into an indivisible group where either all $dt_i \in aut_{m,j}$ commit or abort. Hence, AUT_m is an imposed structure on an *ut*. And, let AUT be the set of all AUT_i .

In contrast to *CNT* which ensures an atomic outcome after the validation takes place, a similar structure is required that coordinates the validation for several *dt*, i.e., their write spheres, grouped in an *aut*. Also, amongst all *aut*. Thus, a transaction structure for the validation is required. Further, the commitment rules and the coordination amongst the *dt* is different to *CNT*. This is caused by the time between the termination of two *dt*. Within one *dt* it is possible to bring in the modifications of one atomic unit of *dt* in one step (there is only one write phase), at this stage, one dt_i may enter its termination (p3,p4) a long time before another dt_k of the same atomic unit enters its termination.

Eventually, this means that isolation is weakened and compensation is required to semantically undo the effects because a *dt* has to commit to release its locks and isolation is no longer possible. This requires, as defined in definition 1, each *dt* to define its compensation, which may lead to a less favourable cascading compensation. We refer to [6], [12], [16] for a thorough discussion about compensation. Isolation is no longer given, but since each *t* and, so each *dt*, must pass the validation, our approach is able to detect and prevent inconsistencies albeit after they arise, which we nevertheless consider as a clear improvement because inconsistencies at the DB level can be avoided despite a weak isolation.

Now, the coordination mechanism on top of the validation is introduced. Similar to *CNT* we introduce an Open Nested Transaction *ONT* which is considered open because of the weak isolation.

Definition 8: Open Nested Transaction *ONT*

- (1) Let $ONT(dt_i)$ be an open nested transaction over ut_i which is defined as a tree $ONT(ut_i) = (Vont, Eont)$ with ut_i as root node r which only commits if all its children commit, the set of vertexes $Vont \equiv AUT$ and the set of edges $Eont$ is defined as $\forall aut_{m,j}, aut_{m,k} \in Vont : aut_{m,j} \rightarrow aut_{m,k} \Leftrightarrow eont \in Eont$ with $eont = (aut_{m,j}, aut_{m,k})$.
- (2) If there exists an order (edge) $\exists eut \in Eut_m = dt_i \rightarrow dt_l$ with $dt_i \in aut_{m,j}$ and $dt_l \in aut_{m,k}$ for $j \neq k$ then there must be a dependency between $aut_{m,j}$ and $aut_{m,k}$ so that $dt_i \rightarrow dt_{first} \in aut_{m,k}$, and $dt_{last} \in aut_{m,k} \rightarrow dt_{i+1} \in aut_{m,j}$, with dt_{first} as the first and dt_{last} as last element in $aut_{m,k}$, and let dt_{i+1} be the successor of dt_i so that $dt_i \rightarrow dt_{i+1}$. Thus $aut_{m,k}$ becomes part of $aut_{m,j}$. If $\neg \exists (dt_i \rightarrow dt_{i+1})$ then $aut_{m,j}$ runs concurrent to $aut_{m,k}$.
- (3) Further, the construction of an order of atomic units must terminate.

Please bear in mind that a single *dt* is also encapsulated by an *aut* and that a *dt* commits if its wsp^w does. So, actually *ONT* is a structure over all wsp^w which are themselves structured by *CNT*. Whereas *ONT* coordinates above the validation, *CNT* coordinates below. To ensure an atomic outcome for a *dt* some further specification for *ONT* is required:

Definition 9: Commitment rules of *ONT*

- (1) A parent is only allowed to commit if all its children commit. If one of its children abort compensation is required.
- (2) Between a chain of *aut* the following holds:
 - (a) A successor of $aut_{m,j}$ is only allowed to start if its predecessor $aut_{m,j-1}$ commits.
 - (b) If one *aut* in the chain aborts all other *aut* have to abort too, and the compensation handler of each *dt* of the same *aut* must be called in reverse order as defined by *Gut*

The commitment rules of *ONT* are different from the ones defined for *CNT* due to the open nature. As a concrete realisation we propose to lodge an execution plan by a coordinator (CO) for each *ut*, hence the model of the *ONT* itself. Each *dt* which enters its write sphere has to register and must (i) await the CO's acknowledgement to enter the validation and (ii) it has to send the final outcome, commit or abort, to the CO. If CO receives a commit message from a *dt* it marks the corresponding node in the graph as committed, and in case of an abort CO does not only mark the node as aborted, it must initialise the compensation too. To control the compensation a compensation model $comp^{-1}(ut)$ must be derived. We omit a definition and the interested reader is referred to [12].

A *dt* is only allowed to enter the validation if all its predecessors have already committed. A parent of several siblings is only allowed to commit if all its children have committed. To release locks only if all siblings commit is not required because of the compensation, hence a 2PC [1] in its classic definition is not required. Following this approach the CO can obey the order and the status of each *dt*. The suggested approach can be interpreted as reducing a graph to its root node.

To complete the model we must show that the concurrent execution of atomic units *aut* is also consistent. The argumentation follows the same way as at the end of section III-B.

IV. RELATED WORK

The Nested Transaction Model has been introduced by Moss [8] and can be seen as the seminal work concerning Advanced, Workflow, or Business Transaction Models (see [2], [4]). Spheres have been introduced by Davies [17] and can be seen as the generalisation of the Nested Transaction Model which itself can be seen as a generalisation of the chained transaction model [1].

Compensation was already foreseen by Moss, but especially the SAGA [6] model heavily applied compensation transactions. The work by Leymann [12] especially focused on the existence of atomic and compensation spheres. One influential workflow transaction model is Reuters Contract model [18]

which is a conceptual framework for the reliable execution of long-lived computations in a distributed environment. OCC was introduced by Kung and Robinson [5] and even if it never gained a lot of attention as a CC mechanism in a DBMS, its validation concept has been applied in synchronisation concepts in Mobile Computing, for example, but rarely adopted into the MW itself. The PyrrhoDB [19] is the only database we are aware of that implements OCC as CC mechanism. Laiho and Laux [20] thoroughly analysed Row Version Verification (RVV) as an implementation of OCC within a disconnected architecture and their work provides, beside a detailed discussion, patterns to implement RVV for a couple of common databases and data access technologies at the MW layer. Fekete et al. [21] also pointed out that an integration of underlying short transactions and complex transactions is important, and mechanisms to ensure consistency without the need to lock data are required. Their work, however, introduces research directions and not a formal model describing the interdependence between flat and complex transactions.

V. CONCLUSION AND FUTURE WORK

The defined transaction model decouples the DB from the MW layer by applying an OCC at the MW layer and imposes an OCC phase model within a component. This enables to detect overall inconsistencies despite temporal sub-transactional inconsistencies. We also interrelated flat transactions with the complex transaction structure by applying a *CNT* after the validation to ensure an atomic outcome, and *ONT* coordinates the atomic outcome of *ut* above the validation. This interrelation closes the gap between flat and complex transactions, and the application of OCC at the MW can also help to establish a loose transaction coupling between DB and MW.

Our future work will include a theoretical underpinning, especially proofs, and focus on an implementation combining our previous work [7] with this work. Our specific focus is thereby on the semantics of transactions, as well as how their transactional requirements can be expressed. On that account we are working on the quantification of consistency requirements. This abstract model will form the basis for our future work.

REFERENCES

- [1] Jim Gray and Andreas Reuter. *Transaction Processing: Concepts and Techniques*. Morgan Kaufmann, 1993.
- [2] Sushil Jajodia and Larry Kerschberg, editors. *Advanced Transaction Models and Architectures*. 1997.
- [3] Ahmed Elmagarmid, Marek Rusinkiewicz, and Amit Sheth, editors. *Management of heterogeneous and autonomous database systems*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.
- [4] Ting Wang, Jochem Vonk, Benedikt Kratz, and Paul Grefen. A survey on the history of transaction management: from flat to grid transactions. *Distrib. Parallel Databases*, 23(3):235–270, 2008.
- [5] H. T. Kung and John T. Robinson. On optimistic methods for concurrency control. *ACM Trans. Database Syst.*, 6(2):213–226, 1981.
- [6] Hector Garcia-Molina and Kenneth Salem. Sagas. In *SIGMOD '87: Proceedings of the 1987 ACM SIGMOD international conference on Management of data*, pages 249–259. ACM, 1987.
- [7] Fritz Laux and Tim Lessner. Escrow Serializability and Reconciliation in Mobile Computing using Semantic Properties. *International Journal On Advances in Telecommunications*, 2(2):72–87, 2009.
- [8] J. Eliot B. Moss. *Nested transactions: an approach to reliable distributed computing*. Massachusetts Institute of Technology, Cambridge, MA, USA, 1985.
- [9] J. Eliot B. Moss. Open nested transactions: Semantics and support. *Workshop on Memory Performance Issues*, 2006.
- [10] Gerhard Weikum and Hans-Jörg Schek. Concepts and Applications of Multilevel Transactions and Open Nested Transactions. In *Database Transaction Models for Advanced Applications*, pages 515–553. Morgan Kaufmann, 1992.
- [11] Gerhard Weikum and Gottfried Vossen. *Transactional Information Systems: Theory, Algorithms, and the Practice of Concurrency Control and Recovery*. Morgan Kaufmann, 2002.
- [12] Frank Leymann. Supporting Business Transactions Via Partial Backward Recovery In Workflow Management Systems. In *BTW*, pages 51–70, 1995.
- [13] Michael Beisiegel et al. Service component architecture (sca) v1.00, 2010-11-08, 2007.
- [14] Olaf Zimmermann, Jonas Grundler, Stefan Tai, and Frank Leymann. Architectural Decisions and Patterns for Transactional Workflows in SOA. In *ICSO '07: Proceedings of the 5th international conference on Service-Oriented Computing*, pages 81–93. Springer-Verlag, 2007.
- [15] Heiko Schultdt, Gustavo Alonso, Catriel Beeri, and Hans-Jörg Schek. Atomicity and isolation for transactional processes. *ACM Trans. Database Syst.*, 27(1):63–116, 2002.
- [16] Heiko Schultdt and Gustavo Alonso and Hans-Jörg Schek. Concurrency Control and Recovery in Transactional Process Management. In *Proceedings of the Eighteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, May 31 - June 2, 1999, Philadelphia, Pennsylvania*, pages 316–326. ACM Press, 1999.
- [17] C. T. Davies. Data processing spheres of control. *IBM Systems Journal*, 17(2):179–198, 1978.
- [18] Andreas Reuter and Kerstin Schneider and Friedemann Schwenkreis. ConTracts Revisited. In Sushil Jajodia and Larry Kerschberg, editors, *Advanced Transaction Models and Architectures*. 1997.
- [19] Malcolm Crowe (University of the West of Scotland). The Pyrrho database management system (<http://www.pyrrhodb.com/>, 2010-11-02), 2010.
- [20] Martti Laiho and Fritz Laux. Implementing optimistic concurrency control for persistence middleware using row version verification. In Fritz Laux and Lena Strömbäck, editors, *The Second International Conference on Advances in Databases, Knowledge, and Data Applications (DBKDA 2010)*, pages 45–50. IEEE Computer Society, 2010.
- [21] Alan Fekete, Paul Greenfield, Dean Kuo, and Julian Jang. Transactions in loosely coupled distributed systems. In *Proceedings of the 14th Australasian database conference - Volume 17*, ADC '03, pages 7–12, Darlinghurst, Australia, Australia, 2003. Australian Computer Society, Inc.

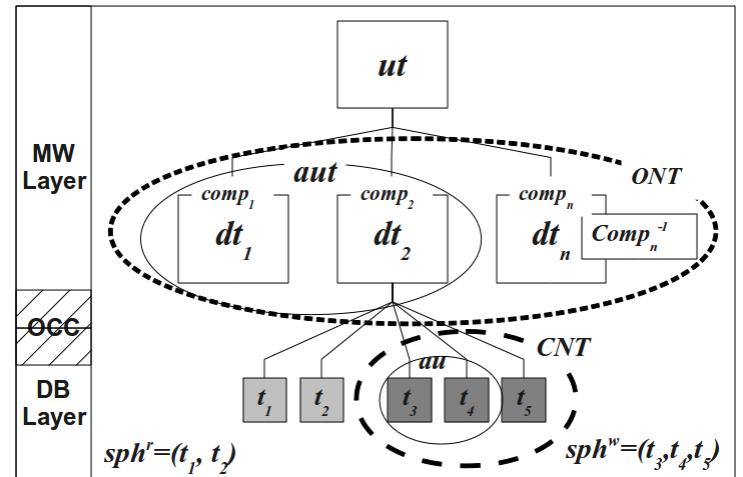


Fig. 2. Dependencies between *ut*, *dt*, *comp*, *t*, *CNT* and *ONT*

A Concept for a Compression Scheme of Medium-Sparse Bitmaps

Andreas Schmidt*† and Mirko Beine*

* Department of Informatics and Business Information Systems,
University of Applied Sciences, Karlsruhe
Karlsruhe, Germany

Email: andreas.schmidt@hs-karlsruhe.de, mirko.beine@arsinventionis.de

† Institute for Applied Computer Science
Karlsruhe Institute of Technology
Karlsruhe, Germany

Email: andreas.schmidt@kit.edu

Abstract—In this paper, we present an extension of the WAH algorithm, which is currently considered one of the fastest and most CPU-efficient bitmap compression algorithm available. The algorithm is based on run length encoding (RLE) and its encoding/decoding units are chunks of the processor's word size. The fact that the algorithm works on a blocking factor, which is a multiple of the CPU word size, makes the algorithm extremely fast, but also leads to a bad compression ratio in the case of medium-sparse bitmaps (1% - 10%), which is what we are mainly interested in. A recent extension of the WAH algorithm is the PLWAH algorithm, which has a better compression ratio by piggybacking trailing words, which look “similar” to the previous fill-block. The interesting point here is that the algorithm also is described to be faster than the original WAH version under most circumstances, even though the compression algorithm is more complex. Therefore, the concept of the PLWAH algorithm was extended to allow so-called “polluted blocks” to appear not only at the end of a fill-block, but also multiple times inside, leading to much longer fill lengths and, as a consequence, to a smaller memory footprint, which again is expected to reduce the overall processing time of the algorithm when performing operations on compressed bitmaps.

Keywords-Compressed bitmaps, WAH algorithm, RLE, CPU-memory-gap

I. INTRODUCTION

Compressed bitmaps play an increasingly important role in efficiently answering multi-dimensional queries in large data sets. Another application is the representation of *positionlists* inside column-stores [1]. We are presently developing a framework with basic components to build column-store applications. Besides *ColumnFile* and *ColumnArray* as basic components, we also identified the *positionlist* as a key component of our framework. A *positionlist* for example is responsible for buffering the data sets that satisfy a condition on a column. This is done by storing a list of tuple-ids. The tuple-ids are sorted and have no duplicates. If the lists are short, tuple-ids can be stored as INT (4) values, but in the case of millions of entries in a *positionlist*, the (compressed) bitmap is the more appropriate representation form. After analysing the relevant scientific papers about

bitmaps, we identified the well-known WAH algorithm [2] as one of the candidates for implementing our *positionlist*. One drawback of the algorithm was that an efficient compression is only possible when the selectivity is about 0.1% and below. In our operational area, however, also selectivities between 1% and 10% should be handled efficiently. A recent extension of the WAH algorithm is the PLWAH algorithm [3], which has a better compression ratio (up to a factor of 2) by piggybacking trailing words, which look “similar” to the previous fill block. The interesting point here is that the algorithm also is described as faster than the original WAH version under most circumstances even though the compression algorithm is more complex. This leads to the assumption that the CPU memory gap [4] has shifted the algorithm from CPU bound to IO bound in the past years and that the bottleneck of the algorithm is no longer the CPU, but the access to the main memory. In this case, processing time may be reduced by finding a better compression for selectivities between 1% and 10%.

The paper is organised as follows. In the next section, we introduce the main concepts of the WAH algorithm. Afterwards, we present our extension of the WAH algorithm, which introduces a new fill type that cannot only handle identical bits in a fill, but also allows for the existence of a number of pollutions inside. Subsequently, we explain our concept using an example and after that, a number of possible variants will be discussed. Our paper will be completed with a short summary and a longer list of activities once the implementation of our algorithm will be available.

II. RELATED WORK

The WAH algorithm is a compression algorithm for bitmaps. It is based on run length encoding and allows for efficient operations on the compressed versions of the bitmaps. It is very CPU-efficient, because it uses the CPU word size as basic packing unit, which allows very efficient operations on the data. Two types of blocks are distinguished. *Literal* blocks contain uncompressed bits and *fill* blocks contain a number of subsequent identical bit values. In the remaining

of the paper, we will focus, without loss of generality, on the 32-bit version of the algorithm. In this case, each *literal* block contains 31 uncompressed arbitrary bits and each *fill* block holds a multiple of 31 bits with the same value. The first bit of a block is used to distinguish a *fill* block from a *literal* block. To separate a *0-fill* from a *1-fill*, the second bit is used. 30 bits are left to indicate the length of a fill. The length is given in multiple of 31 bits, not in individual bits. A value of 2 means a fill with 62 identical bits. Figure 1 shows the compression of a bitmap of 194 bits length (first line). First, the bitmap is divided into equidistant parts of 31 bits (second line) and these parts are further classified as *fill* or *literal*. After that, consecutive fills with the same bit value are combined.

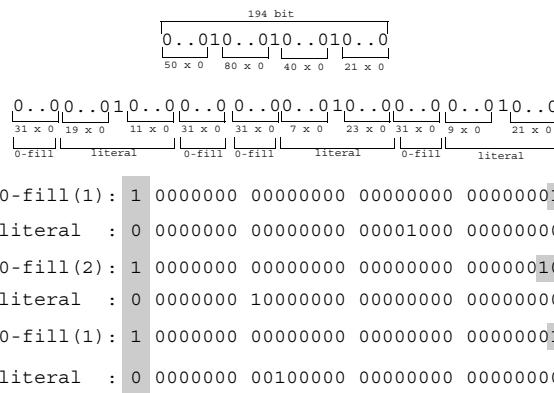


Figure 1. Bitmap compression with WAH

The drawback of this algorithm is that in the case of medium-spare bitmaps, the *fills* are very short and every single “pollution”, leads to a full literal block. The switch between a fill and a literal (and back to a fill) block is an expensive job in terms of memory.

III. CONCEPT

The main difference between WAH and our concept is that we support the concept of *dragged fills*, which allows a small number of false bits inside each word of a fill. The intention here is to obtain longer fills, because the switch from a fill to a literal block and back to a fill is an expensive act in terms of memory. The PLWAH (position list word aligned hybrid) method uses a related concept by piggybacking a trailing literal block after a fill, if it differs from the words in the preceding fill by one bit only. For this purpose, the length field in the fill is reduced by some bits, while five of these bits indicate the position of the wrong bit in the trailing literal. With this trick, you can achieve a reduction by a factor of two for certain distributions of data. Otherwise, the maximum length of a fill is reduced by a factor of 2^6 , may reach a maximum of 2^{24} instead of 2^{30} .

In contrast to this, our concept does not only allow for one slightly polluted literal at the end of a fill, but it also

allows for slightly polluted literals to appear at each position in the fill without reducing the overall length of a fill.

A. Dragged Fill

Our concept requires the introduction of a new block type called *dragged fill*, which can handle the polluted literals inside a fill. In contrast to the other two block types *literal* and *fill*, a *dragged fill* has a variable length depending on the number of pollutions inside. Hence, three different types of blocks (literal, fill, and dragged fill) must be distinguished. We distinguish a *fill* from a *dragged fill* with the third significant bit, so that a 1-fill is identified by the bit combination of 111, while a dragged-1-fill is identified by 110 (0-fill: 101, dragged-0-fill: 100). The indicator of a *literal* remains identical to the WAH algorithm (a 0-bit at the most significant bit), which still allows us to store 31 bits in each *literal*. For every word in a *dragged fill*, we first have to define how polluted it could be to be part of such a fill. For a 32-bit version of the WAH algorithm, different degrees of pollution can be defined, which vary from one wrong bit inside 32, 16, 8, and 4 bit, leading to 1, 2, 4, or 8 wrong bits (called pollution factor) in a complete 32-bit word¹. Figure 2 presents examples of different pollution factors, each with the maximum number of skipped bits.

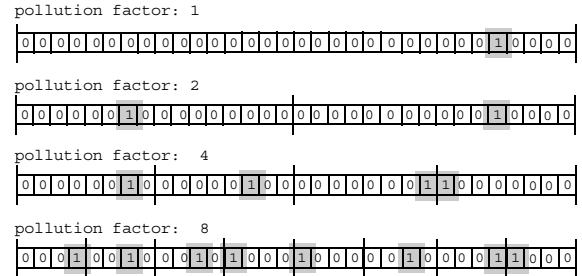


Figure 2. Possible pollution factors for a block

Each polluted 32-bit word needs a fixed number of bits for description. The value of needed bits is dependent on the pollution factor and the maximum length of a fill. In case of a pollution factor of 1, we only need to specify the position of the wrong bit, which could be done with 5 bits ($2^5 = 32$). With a pollution factor of 2, we need 4 bits to specify the position of the wrong bit in the first 16 bits and another 4 bits to specify the wrong bit in the second half of the word. As only one of the two 16-bit words may contain a wrong bit, we need a mask of another 2 bits to specify in which part the skipped bits occur. Table I gives an overview of the memory consumption also for the other pollution factors.

Additional memory is needed to specify the position of the polluted words. The size is dependent on the maximum

¹Strictly speaking, we do not have 32 bits, but only 31 bits as packing unit. But for the sake of straightforwardness in explaining the concept we talk in this paper about 32 bits. Keep in mind that, without loss of generality, one bit can be ignored, i.e. the leftmost one.

Table I
MEMORY CONSUMPTION OF DIFFERENT POLLUTION FACTORS

Pollution factor	Memory consumption (in bit)
1	5
2	10 ($2 * 4 + 2$)
4	16 ($4 * 3 + 4$)
8	24 ($8 * 2 + 8$)

length of a fill. If for example the maximum value is 1024 (2^{10}), 10 additional bits are required to specify the position for each polluted 32-bit word in the most simple implementation, where the position is specified by its index inside the run. Later in section III-C, we will discuss different possibilities to identify the wrong words.

B. Example

After the introduction of the concept, the effect will now be demonstrated using the example given in Figure 3.

In the middle part of the Figure, seven 32-bit blocks can be seen. Except for the fourth and the sixth block, which contain two and one polluted bit(s) (indicated in grey) all blocks contain 0-bits only. The two polluted blocks are shown in detail in the upper and lower part of the Figure. The pollution factor is set to 2, meaning that we can accept one wrong bit in every 16-bit of the block at the most. So both polluted words can be accepted to be inside a *dragged fill* and the overall length of the fill is 7 words. Besides the overall length, we have to provide additional information for a *dragged fill*. This information includes:

- The number of polluted blocks
- The positions of the polluted blocks inside the fill
- Position of the wrong bits inside a polluted block

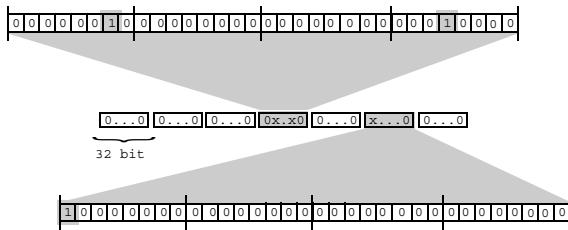


Figure 3. *Dragged fill* with two polluted blocks

The maximum number of polluted blocks depends on the maximum length of a *dragged fill* and the number of bits to specify the number. The same holds for the specification of the position of the polluted blocks. In our example, we choose a maximum length of a *dragged fill* of 64 and a pollution factor of 2. This means that we need 6 bits to specify the size of the fill and another 6 bits to specify the number of polluted blocks inside the fill. For each polluted block, we also have to provide the information on the position of the block inside the fill and the wrong bits

inside. Figure 4 shows a possible memory layout for the above example in the upper part. The first three bits are reserved for the block type, then 6 bits for the fill length field, and another 6 bits for the field indicating the number of polluted blocks.

In the lower 16 bits of the first word, the information about the individual polluted words inside a fill is contained. In the defined layout (maximum length: 64, pollution factor: 2), we need exactly 16 bits to specify one pollution word. The first two bits, labeled as “mask”, identify in which of the two 16-bit words a pollution occurs. Possible values are 01, 10, and 11. The next 6 bits specify the position of the polluted word inside the fill. As a maximum of 1 wrong bit can occur inside one 16-bit word, we need 4 more bits to specify the position (0..15) of the wrong bit inside a 16-bit word. As we have two 16-bit words in our polluted block, we need another 4 bits for the second word. In each following 32-bit word, we can now store the information of two more polluted words.

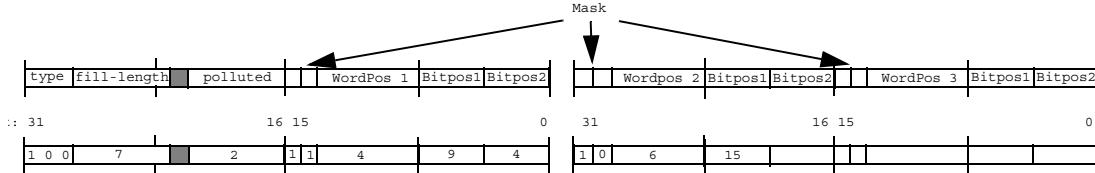
In the lower part of Figure 4, the true values for the example in Figure 3 are presented. First, the block type for a *dragged-0-fill* is specified, followed by the information of a fill length of seven with two polluted words. Then, the ‘11’ mask indicates, that there are two skipped bits in the polluted block at position 4 in the fill. The two skipped bits can be found at bit-position 9 (first 16-bit word) and bit-position 4 (second 16-bit word), respectively. In contrast of this the second polluted block only contains one wrong bit in the first 16-bit word (mask ‘10’), which can be found at position 15.

The total memory footprint is 64 bits, compared to 160 bits in the original WAH implementation² and 128 bits in the PLWAH implementation. Especially in cases of lower selectivity, the proposed concept is superior with regard to memory footprint. The high memory cost of switching from a fill to a literal block and back can be avoided in many cases. And even in the case where no fills can be found, there is no drawback due to the fact that a literal block can handle 31 bits as in the original WAH-algorithm. One little drawback exists in the case of a very high selectivity leading to extremely long fills: Because of the new block type, the proposed concept needs one bit more to indicate a fill block, and so a block can contain a maximum of $2^{29} * 31$ bits instead of $2^{30} * 31$. As our concept has not been yet implemented, we cannot make any statements about the runtime behaviour. However, we plan to run a bunch of experiments with different data distributions concerning runtime and memory behaviour.

C. Variants

In the above concept we divided each 32-bit block into equidistant parts, which can contain 1 wrong bit at the most.

²160 bits = 32 bits (0-fill, length: 3) + 32 bits (literal word) + 32 bits (0-fill, length: 1) + 32 bits (literal) + 32 bits (0-fill, length: 1)

Figure 4. Memory layout of a *draggled-0-fill*

This solution was chosen, because it is easy to implement and also CPU-efficient.

Another, more general solution may be not to divide the block into equidistant parts, but to allow a maximum of n -skipped bits to appear inside a 32-bit block. In this case, the memory consumption is a little bit higher, but it is a more general model, which can lead to longer fills.

Instead of specifying the index position of a polluted block, it is also possible to specify the gaps between polluted blocks (incremental encoding [5]). This leads to a smaller memory footprint for each polluted block, because a lower number of bits can be used to specify the increments. In case the next polluted block is too far away to code the distance with the chosen number of bits, the fill has to terminate. Figure 5 gives an example of this encoding. Each gray square represents a 32-bit block (with unique values, polluted and mixed). The full length of the fill is 21 blocks. As you can see, the values of the increments remain small in contrast to the index encoding in the last line, thus allowing for a lower number of bits to encode the fill.

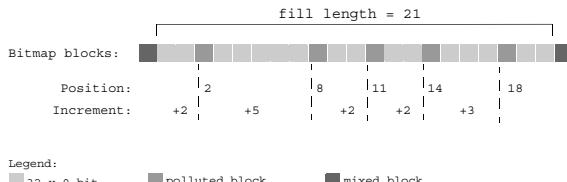


Figure 5. Incremental encoding of “polluted blocks”

All of the above variants require a predefined fixed number of bits to encode the position of the polluted blocks. Another possible solution would be to use a Rice (Golomb) coding [6]. The idea behind this coding scheme is to use a flexible number of bits to encode arbitrarily long integer numbers. Small, but frequently appearing numbers only need a small number of bits, while unfrequent big numbers need more bits as in a normal coding scheme.

IV. CONCLUSION

We presented an extension of the WAH algorithm, which is currently considered one of the fastest and most CPU-efficient compression techniques for bitmaps. However, in the case of a selectivity of 1% and more, the compression behaviour is unsatisfying. The reason for this behaviour is the blocking factor of 32, which requires packing of a

minimum of 31 bits. Even a single skipped bit leads to a literal block, which holds 31 uncompressed bits.

Our contribution handles this problem by allowing so-called polluted blocks to be part of a fill. A polluted block is a block, which has a limited number of wrong bits. The idea is to describe the position of the polluted block and the wrong bits inside it, which takes much less memory than ending a fill, starting a new literal block, and after that starting a new fill.

V. FUTURE WORK

Currently, we don't have an implementation of our concept, but we are working on it. At the time we have a our implementation finished, we plan a number of tests with different selectivity, both synthetical and real world data, comparing both the compression ratio and the execution time of the different operations. Depending on the results we eventually implement different variants of our algorithm, discussed in III-C. Another interesting point would be to look for dependencies between the pollution factor and the maximum fill-length for different data sets.

REFERENCES

- [1] D. J. Abadi, S. R. Madden, and M. Ferreira, “Integrating compression and execution in column-oriented database systems,” in *SIGMOD*, Chicago, IL, USA, 2006, pp. 671–682.
- [2] K. Wu, E. J. Otoo, and A. Shoshani, “Compressing bitmap indexes for faster search operations,” in *SSDBM '02: Proceedings of the 14th International Conference on Scientific and Statistical Database Management*. Washington, DC, USA: IEEE Computer Society, 2002, pp. 99–108.
- [3] F. Deliège and T. B. Pedersen, “Position list word aligned hybrid: optimizing space and performance for compressed bitmaps,” in *EDBT '10: Proceedings of the 13th International Conference on Extending Database Technology*. New York, NY, USA: ACM, 2010, pp. 228–239.
- [4] S. Manegold, P. A. Boncz, and M. L. Kersten, “Optimizing database architecture for the new bottleneck: memory access,” *The VLDB Journal*, vol. 9, no. 3, pp. 231–246, 2000.
- [5] I. H. Witten, A. Moffat, and T. C. Bell, *Managing gigabytes (2nd ed.): compressing and indexing documents and images*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999.
- [6] S. W. Golomb, “Run-length encodings,” *IEEE-IT*, vol. IT-12, pp. 399–401, 1966.

Merging Differential Updates in In-Memory Column Store

Jens Krueger, Martin Grund, Johannes Wust, Alexander Zeier, Hasso Plattner

Hasso Plattner Institute for IT Systems Engineering

University of Potsdam

Potsdam, Germany

jens.krueger@hpi.uni-potsdam.de, martin.grund@hpi.uni-potsdam.de, johannes.wust@hpi.uni-potsdam.de
alexander.zeier@hpi.uni-potsdam.de, hasso.plattner@hpi.uni-potsdam.de

Abstract—To meet the performance requirements of enterprise application for both, transactional application as well as analytical scenarios, data storage of in-memory databases are split into two parts: One optimized for reading and a write-optimized differential buffer. The read-optimized main storage together with the differential buffer for inserts provide the current state of the database. In regular intervals the differential buffer is merged with the main database to maintain compression and query performance. This merge process runs asynchronous to minimize the impact on query performance. However, simple duplication of the data structures prior to the merge process lead to a main memory consumption of at least twice the size of the database. In this paper we propose a differential merge update based on single columns. In typical enterprise application data environments this leads to a significant reduction of memory consumption as this type of applications tend to store transactional data in very large single tables. The Single Column Merge has been implemented in HYRISE and proved in a test scenario based on real enterprise data.

Index Terms—In-Memory Database; Column Store; Merge Process;

I. INTRODUCTION

Enterprise data management systems currently in use are typically being optimized either for transactional data processing (OLTP) or analytical data processing (OLAP). In order to combine both requirements for mixed workload scenarios the introduction of a write optimized differential buffer together with a read-optimized main storage has been proposed in [4], [8], [14]. The main advantage of this design is that the compression of the read storage does not need to be re-compressed every time a data modification operation is executed as all changes are stored in a differential buffer. However, the main storage and the differential buffer have to be merged at some point of time to maintain the performance in read intensive scenarios, mainly for two reasons:

- Merging the differential buffer into the main relation decreases the memory consumption since better compression techniques can be applied.
- Additionally, merging the buffer allows better read query performance due to an order-preserving value dictionary of the main store.
- Furthermore, the bit compression of valueID's allows better bandwidth utilization which leads to improved read

performance since in-memory databases suffer from the bandwidth limitations of todays hardware.

The key requirement for the merge process is to have as little impact as possible on the performance of the database. Therefore it has to run asynchronously to other operations such as query execution. The cost of this process is mainly determined by the performance impact on the other operations and main memory consumption. This paper focuses on the optimization of the memory consumption.

A. Enterprise Application characteristics

We applied the concept of a differential buffer to column-oriented, in-memory databases, as we could show that these databases perform especially well in Enterprise Application scenarios. By analyzing customer applications and customer data we derived typical enterprise application characteristics as shown in [9], [10]. The most important findings based on the customer system analysis and their implications on database design are:

- Enterprise applications typically present data by building a context for a view, modification to the data only happen rarely. Hence column-oriented, in-memory databases that are optimized for reading as proposed in [8], [12] perform especially well in enterprise application scenarios. In fact, over 80% of the workload in an OLTP environment are read operations.
- Tables for transactional data typically consist of 100-300 columns and only a narrow set of attributes is accessed in typical queries. Column-oriented databases benefit significantly from this characteristic as entire columns, rather than entire rows, can be read in sequence.
- Enterprise data is sparse data with a well known value domain and a relatively low number of distinct values. Therefore data of enterprise applications qualifies very well for data compression as these techniques exploit redundancy within data and knowledge about the data domain for optimal results. Abadi et al. have shown in [1] that compression applies particularly well to columnar storages. Since all data within a column a) has the same data type and b) typically has similar semantics and thus low information entropy, i.e. there are few distinct values in many cases.

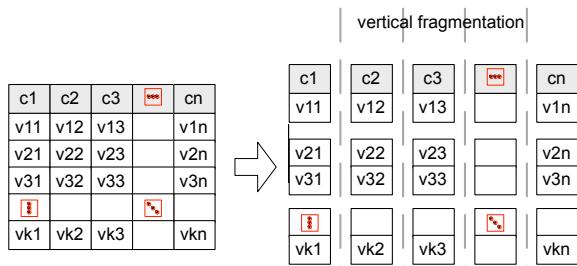


Fig. 1. Column-oriented storage paradigm

- Enterprise applications typically reveal a mix of OLAP and OLTP characteristics [10]. To support both, the data storage of in-memory databases are split into two parts, one optimized for reading and one for writing.
- Data growth in enterprise systems has not shown the same growth rate as for example social networks. Despite the fact of a growing number of captured events in enterprise environments all events are based on actual events related to the business which have an inherent processing limit by the size of the company.

Given that findings on enterprise application our approach to build an application-specific data management is focused on in-memory data processing with data compression and column-wise data representation in order to utilize todays hardware as best as possible.

B. Structure of the paper

The remainder of the paper is structured as the following: First we introduce HYRISE, our prototypical column-oriented, in-memory database prototype used to empirically validate our findings. The next section gives an overview of the traditional merge process of main storage and differential buffer. In Section IV, we propose a modified merge algorithm, the Single Column Merge, that reduces additional memory consumption during the merge process. Section V gives an overview of related work while Section VI concludes this work.

II. OVERVIEW OF HYRISE

A. HYRISE architecture

The following section describes the architecture of the HYRISE prototype, including the storage manager and query executor.

The storage manager maintains the physically stored data in main memory and provides access methods for accessing data while organizing data along columns with applied dictionary compression. Consequently, all relations are fully decomposed while a surrogate identifier allows the reconstruction of tuples of the column partitions. Figure 1 shows the vertical fragmentation of a table as used in HYRISE and depicts that attribute focused read operation can exploit sequential memory access while tuple reconstruction requires random access to each column. By choosing to optimize this database prototype for an online mixed workload (OLXP) as described in [10]

the reconstruction of complete relation in a timely manner gets equally important as data modifications and scans over large sets of data.

In case of HYRISE the row or surrogate identifier is implicit and can be extracted from the position of a value in a column. Therefore, fast access due to offsetting is made possible which can also be leveraged in positional joins algorithms. Unlike other lightweight compression techniques the implemented dictionary encoding enables this positional access since it facilitates the change of variable-length fields into fixed-length data types on each column.

In order to speed up read access by as late as possible decompression of the actual value the dictionary of the encoding in HYRISE is sorted leading to order-preserving values in the actual column. Considering this, predicates can be applied on the attribute vector and ranges can be looked up without decompressing every single value. Besides, the sortation enables fast binary search on the dictionary.

Furthermore, the storage bit-compresses the values pointing from the dictionary to the attribute vector by using only the amount of bits necessary to represent the cardinality of distinct values of each column. Especially in enterprise applications the attributes are characterized by a limited domain. Hence, bit compressing value identifiers is very effective and improves the compression factor even more. Besides the additional compression bit compressed value identifiers support better bandwidth utilization in late materialized query executions.

While this extended dictionary compression technique offers both good compression ratio and optimized read access modifications of data are almost impossible due to fact that the data would have to be re-compressed every time modification operation would be executed. For example, if a new value would change the sort order of the existing dictionary or the cardinality of distinct values changes in a way that the already used bits are not sufficient the complete attribute vector has to be modified. Consequently, all modifications are handled by a dedicated differential buffer for each table to postpone re-compression cost to later point of time to distribute the re-compression cost over all data modifications stored in the buffer. This re-compression is done by merging differential buffer and main storage. The buffer implements a vertical partitioning as well but leaves out both the order-preserving and value bit-compressing optimizations in order to allow fast appends to the table. This architecture is based on the fact that decomposition of relations in main memory with the lookup or extension of the dictionary is way faster than writing the log to disk that has to happen in in-memory databases to assure durability.

Given the fact of a dedicated buffer to handle all data changes, updates have to be implemented as an insert followed by an invalidation of the to be updated record. The invalidation is maintained by a two bit vectors, which keep track of updates and deletes in the compressed storage and the corresponding differential buffer. The storage manager is in charge of keeping data consistent what in this case means the main storage and delta storage have to be kept in sync and corresponding merge

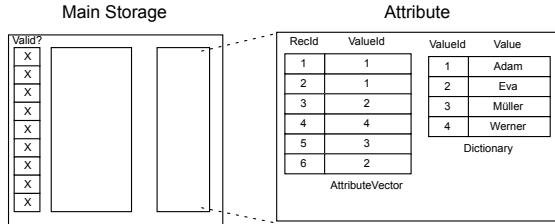


Fig. 2. Example of a main storage

processes have to run asynchronously to avoid conflicts with running queries during the merge process.

The query executor is responsible for executing a given query plan, including loading the necessary meta data and materializing results with regards of late materialization strategies that are a result of the column-wise data representation with applied dictionary encoding. The current state of the system provides no direct access using a query language like SQL but focuses on the implementation of the plan operators. It implements the necessary relational algebra operators and leaves the query plan design up to the user of the prototype. Hence the query plans used are written by hand and than executed by the execution engine while assuming that this written query plans are optimal and no further optimization takes place.

For the purpose of this study, some features of a conventional database such as multi-threading, transactions, or recovery are not implemented to avoid the related overhead. We omit these features because we believe they are orthogonal to the question of how to compact data using a merge process in an in-memory column store. For the same reason the process of loading data from a storage system at startup time is not taken into account.

B. HYRISE data structures

In the following we illustrate the data structures for main storage and differential buffer. Figure 2 shows an illustration of a main storage. The data structures for one column are illustrated in detail. The table *AttributeVector* shows the vector holding the values for each record of a particular attribute. The values are dictionary compressed; therefore the stored *ValueIds* are references to the table *Dictionary* containing the actual values. The *Valid?BitVector* indicates whether this record is still valid or has been invalidated by an update or delete in the differential buffer.

New entries are stored in a write optimized differential storage as shown in Figure 3. The example shows 4 newly added entries in the *AttributeVector*. Similar to the main storage, the differential buffer has a *Dictionary*. The main difference between both storages is the implementation of the dictionary as discussed in the section above. All dictionaries used by the main storage need to be sorted in order to allow binary search and are bit-compressed. In contrast, the dictionaries in the differential buffer are unsorted and not bit-compressed to allow fast appends.

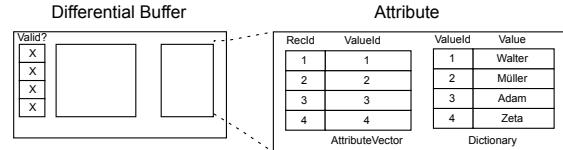


Fig. 3. Example of a differential buffer

III. THE MERGE PROCESS

A. Description of the merge process

The merge process and its complexity is described in detail in [8]; we give a brief overview here. The process can be separated into three phases - *prepare merge*, *attribute merge* and *commit merge*.

The prepare merge phase locks the differential buffer and main storage and creates a new empty differential buffer storage for all new inserts, updates, and deletes which occur during the merge process. Additionally the current valid vector of the old buffer and main storage at merge time are copied to be used throughout the merge process, as these may be changed by concurrent updates or deletes applied during the merge while affecting records involved in this process. In the attribute merge phase the following steps are executed for each attribute: the first step is merging the dictionaries of the differential buffer and main storage. Next, the value ids of main storage and write buffer are copied to a new main storage - thereby changes in the dictionary have to be applied to the new value ids; invalidated values of the original main storage are not copied and can be transferred to a history log. To ensure persistency, the merge result is written to secondary storage.

The commit merge phase starts by acquiring a write lock of the table. This ensures that all running queries are finished prior to the switch to the new main storage including the updated value IDs. Then, the valid vector copied in the first phase is compared to the actual vector to mark potentially invalidated rows - they are eventually deleted in the next merge process. As last step the new main storage replaces the original differential buffer and main storage and the latter ones are unloaded from memory.

Figure 4 shows the result of the merge process based on the differential buffer and main storage shown in figures 2 and 3. New *AttributeVector* now holds all value records of the original main storage, as well as the differential buffer. Note that the new dictionary includes all values from the main and differential buffer and is resorted to allow binary search and late materializing range queries. Therefore the *ValueId* of single value records has changed compared to the original entry in the main storage and differential buffer.

B. Memory consumption of the merge process

As discussed in [8] prior to the commit merge phase the complete new main storage is kept inside main memory. Hence, at this point double the size of the original main storage

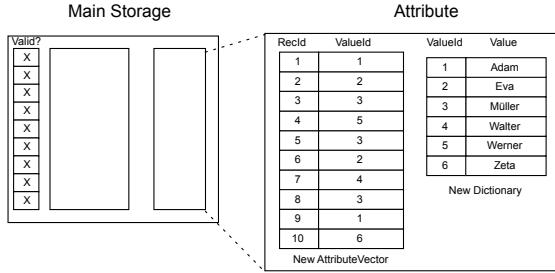


Fig. 4. Example of a delta storage

plus differential buffer is required in main memory to execute the proposed merge process. In the subsequent section we propose a modification of the algorithm to decrease the overall additional memory consumption.

IV. SINGLE COLUMN MERGE

In this section we describe a modified merge process called *Single Column Merge* with the objective of reducing the size of memory consumption throughout the merge process. By merging single columns independently from the delta into the main storage the algorithm reduces the additional memory consumption to the size in memory of the largest column. In order to use this technique the insert only strategy has to be used otherwise records would be physically deleted what could lead to inconsistent surrogate identifiers if merged columns are applied independently. So far deleted records are kept as invalid in the storage system but could be removed by a dedicated garbage collection run.

A. Description of the Single Column Merge

In the merge process described in section III-A the merge result for single columns is calculated independently in the respective attribute merge phases. The merge result is kept in main memory until all attributes are merged to ensure an instant switch to the new main storage in the commit merge phase. The basic idea of Single Column Merge is to switch to an updated main storage after every attribute has been merged while maintaining a consistent view on the data.

Partial hiding of merge results: Switching already merged columns leads to a problem: Some attributes are already merged while others are not. Those finished attributes typically have a longer attribute vector since new rows could have been inserted into the differential buffer. And as this buffer is not updated throughout the merge process value entries for newly created rows are duplicated in the update main storage and original differential buffer. To resolve this issue all newly created rows are marked as invalid until all columns are merged as shown in Figure 5.

Remapping old value IDs: After one attribute is merged, its state differs from the rest of the index that has yet to be merged. Some values potentially have new value IDs if the merge process has changed the value IDs. Incoming queries

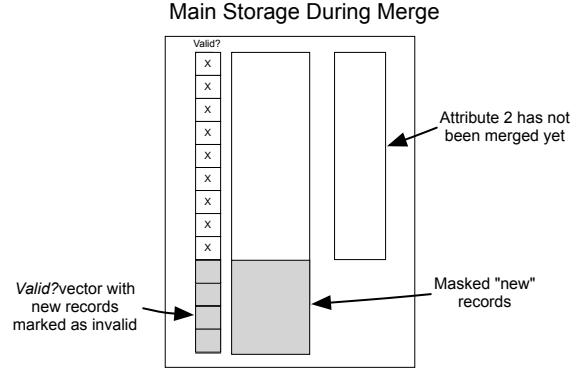


Fig. 5. During the merge: abstract view on the main storage with a single merged attribute

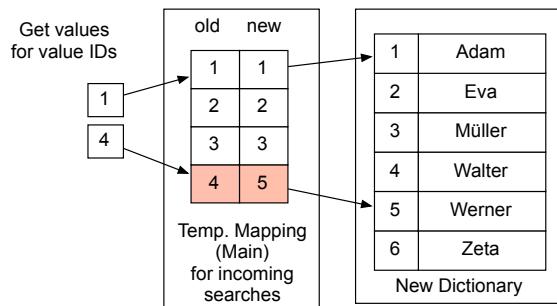


Fig. 6. Example of a remapped lookup for values 1 and 4

might still rely on old value IDs, e.g. in case they have been cached by queries started prior to the merge process. To avoid locking of the table for each attribute a mapping table from the old value IDs to the new ones is provided throughout the merge process until all attributes are merged into the main store. This mapping table from old to new values is created in the attribute merge phase of the merge process described in section III-A when merging the dictionaries of differential buffer and main store. Figure 6 shows an example for a remapped lookup of the cached old value IDs 1 and 4.

Modifications of the traditional merge process: To implement the Single Column Merge as described we have to make the following changes to the merge process as described in section III-A:

- *prepare merge*
 - The valid vector of the main store has to be enlarged by the number of rows that are currently in the differential buffer. This is required to hide the newly created merge results in the main storage until all attributes are merged.
 - The newly created valid record entries are initialized with *false* to deactivate those rows.
 - *attribute merge*: For each attribute the following changes have to be made:

- Keep the mapping tables from old to new value IDs in memory. These tables have to be provided to functions in the query executor that might be called while a merge is running to have a consistent view on the data.
- Switch the attribute data structure of the old main storage to the merge result right after merging the attribute.
- *commit merge*
 - activate the newly merged rows by setting the valid vector entries to *true*.
 - Unload mapping tables from old to new value IDs after the lock on the table is acquired.

B. Evaluation of memory consumption

Applying Single Column Merge eliminates the need to additionally hold the newly created main storage of the size of the original main storage and differential buffer in main memory. As only one attribute is merged at a time the additional amount of main memory needed for the merge process is the size of the attribute data structure currently merged plus the size of the mapping tables from old value IDs to new value IDs for the dictionaries as described in section IV-A. Assuming that the main storage is significantly larger in size than the differential buffer, the overall additional memory consumption for the merge process is driven by the size of the largest data structure of all attributes.

To test how large the savings in additional memory consumption are, we compared the traditional merge process described in section III-A and the Single Column Merge using live customer data. The two major tables in the database consist of 28 million rows with 310 columns and 11 million rows with 111 columns. The main memory usage during the test is shown in Figure 7. The graph shows the additional memory consumption during a merge process for both merge strategies. The column that consumes the most memory can be seen in both test series. The main memory usage during the Single Column Merge clearly peaks at around the size of the largest column, as opposed to the steadily increasing memory usage during the traditional merge.

V. RELATED WORK

Vertical partitioned databases as HYRISE have been researched from the very first conferences on database systems [2], [11], [11], [15] while focusing on read-intensive environments. Pure vertical partitioning into a “column-store” has been a recent topic of interest in the literature. Copeland and Khoshafian [5] introduced the concept of a Decomposition Storage Model (DSM) as a complete vertical, attribute-wise partitioned schema, which has been the foundation for multiple commercial and non-commercial column store implementations such as MonetDB/X100 [4], C-Store [14] or Sybase IQ [7]. All of those examples has shown ability to outperform conventional databases in read-mostly analytic-style scenarios with low selectivity. However, unlike HYRISE, most of the column-store implementations are pure disk based approaches

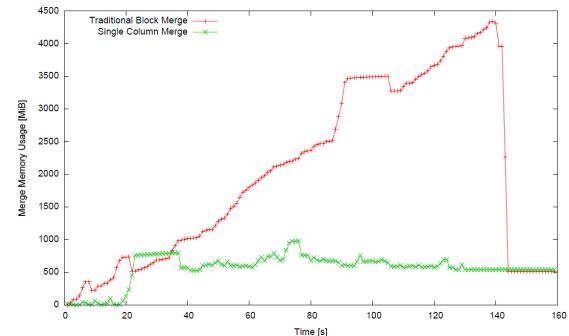


Fig. 7. Main memory usage during traditional merge process and single column merge

and focus to improve the overall performance by reducing the number of disk seeks by decomposing relations. Consequently, data modifications must be propagated to multiple files on disk, which leads to the fact that this implementation variant is inappropriate for workloads combining transactional- and analytical-style queries, because updates and inserts are spread across different disk locations.

As in HYRISE, data compression can limit the applicability to scenarios with frequent updates leading to dedicated delta structures to improve the performance of inserts, updates and deletes. The authors of [4] and [13] describe a concept of treating vertical fragments as immutable objects, using a separate list for deleted tuples and uncompressed delta columns for appended data while using a combination of both for updates. In contrast, HYRISE maintains all data modification of a table in one differential buffer and keeps track of invalidation with a valid bit-vector. However, none of before mentioned work describes in detail how the merge process works.

In contrast to this disk based research, HYRISE builds up on in-memory data processing, which has been influenced in the last decade by the work around MonetDB [3]. The widening gap between the growth rate of CPU speed and memory access speed leads to the usage of compression techniques requiring higher effort for de-compression. Besides the direct effect of storage savings, less physical data has to be transferred from main memory traded for higher CPU costs at the de-compression of the data as described for instance in [16] or [6]. All works on compression on databases systems focus on the data amount reductions and at the same time on query optimizations.

VI. CONCLUSION

Optimized for main memory consumption, the Single Column Merge removes the need to keep a complete copy of the table during the merge process. Instead the main memory consumption can be reduced to a copy of each attribute. The maximum table size increases from half of the total available main memory to the total available main memory minus the

largest columns size. As the merge process is a background task for an operational system queries can still process the data, and lookup information in both the attribute vector and the value dictionary. This concurrency is a requirement for reengineering the merge process in an online mixed workload environment. The Single Column Merge solves concurrency issues by storing an additional mapping table for each column. Every value dictionary lookup during the merge has to access the mapping table first, before it can access the value dictionary. Consequently, this remapping results in one additional random memory access for every value ID lookup but only in case the merge process has not been finished.

REFERENCES

- [1] D. Abadi, S. Madden, and M. Ferreira. Integrating compression and execution in column-oriented database systems. In *SIGMOD*, pages 671–682, New York, NY, USA, 2006. ACM.
- [2] S. Agrawal, V. R. Narasayya, and B. Yang. Integrating Vertical and Horizontal Partitioning Into Automated Physical Database Design. In *SIGMOD*, 2004.
- [3] P. A. Boncz, S. Manegold, and M. L. Kersten. Database Architecture Optimized for the New Bottleneck: Memory Access. In *VLDB*, 1999.
- [4] P. A. Boncz, M. Zukowski, and N. Nes. MonetDB/X100: Hyper-Pipelining Query Execution. In *CIDR*, 2005.
- [5] G. P. Copeland and S. Khoshafian. A Decomposition Storage Model. In *SIGMOD*, 1985.
- [6] G. V. Cormack. Data Compression on a Database System. *Commun. ACM*, 28(12):1336–1342, 1985.
- [7] C. D. French. “One Size Fits All” Database Architectures Do Not Work for DDS. In *SIGMOD*, 1995.
- [8] J. Krueger, M. Grund, C. Tinnefeld, H. Plattner, A. Zeier, and F. Faerber. Optimizing Write Performance for Read Optimized Databases. In *DASFAA*, 2010.
- [9] J. Krueger, M. Grund, A. Zeier, and H. Plattner. Enterprise Application-specific Data Management. In *EDOC 2010*, 2010.
- [10] J. Krueger, C. Tinnefeld, M. Grund, A. Zeier, and H. Plattner. A case for online mixed workload processing. In *DBTest*, 2010.
- [11] S. B. Navathe, S. Ceri, G. Wiederhold, and J. Dou. Vertical Partitioning Algorithms for Database Design. *ACM Trans. Database Syst.*, 9(4), 1984.
- [12] H. Plattner. A common database approach for oltp and olap using an in-memory column database. In *SIGMOD*, 2009.
- [13] R. Ramamurthy, D. J. DeWitt, and Q. Su. A Case for Fractured Mirrors. In *VLDB*, 2002.
- [14] M. Stonebraker, D. J. Abadi, A. Batkin, X. Chen, M. Cherniack, M. Ferreira, E. Lau, A. Lin, S. Madden, E. J. O’Neil, P. E. O’Neil, A. Rasin, N. Tran, and S. B. Zdonik. C-Store: A Column-oriented DBMS. In *VLDB*, 2005.
- [15] P. J. Titman. An Experimental Data Base System Using Binary Relations. In *IFIP Working Conference Data Base Management*, 1974.
- [16] T. Westmann, D. Kossmann, S. Helmer, and G. Moerkotte. The Implementation and Performance of Compressed Databases. *SIGMOD Record*, 29(3), 2000.