A performance evaluation study for QoS-aware web services composition using heuristic algorithms

Pedro F. do Prado
ICMC - USP
Sao Carlos, SP - BR
pfprado@icmc.usp.br

Luis H. V. Nakamura
ICMC - USP
Sao Carlos, SP - BR
nakamura@icmc.usp.br

Julio Estrella
ICMC - USP
Sao Carlos, SP - BR
jcezar@icmc.usp.br

Marcos J. Santana
ICMC - USP
Sao Carlos, SP - BR
mjs@icmc.usp.br

Regina H. C. Santana
ICMC - USP
Sao Carlos, SP - BR
rcs@icmc.usp.br

***Abstract*—In this paper five different algorithms are proposed to solve the QoS-aware Web Services Composition (QWSC) problem in ten different search-space sizes and a realistic deadline (a point not covered in many related works). Differently from some related works, statistical techniques are adopted in this paper to ensure more precise results from the algorithms. The results obtained showed that the design of experiments and the performance evaluation can be used to determine which algorithms have better performance according to the different search-space sizes and the estabilished deadline; it is also possible to determine which genetic operators are better suited for the QWSC problem.**

***Keywords*-qos-aware web services composition; performance evaluation; heuristic algorithms; e-commerce;**

I. Introduction

Nowadays, QWSC is one of the most interesting research issues on Service-Oriented Architecture (SOA). Actually, it is not a new research issue and a Genetic Algorithm (GA) was proposed to solve this problem in 2005 by Canfora et. al. (2005) [1]. In that paper, an empirical study compared a GA with Integer Programming (IP) based algorithm. The results proved that GA was better suited for the QWSC problem. Other related works also compared IP-based algorithms with GAs concluding that a GA is a better alternative [2]. Furthermore, recent works also used GAs or hybrid algorithms (GA combined with another technique) to solve this problem [3] [4]. QWSC is a combinatorial NP-Hard problem so it is a complex problem to solve. The number of possible composition plans (the size of the search-space) grows exponentially according to the size of the composite plan. Thus, the use of Exhaustive Search (ES) algorithms or numerical method algorithms is limited to only very small search-space sizes. In addition, these algorithms become more and more obsolete when the growing proliferation of the use of Web Services (WS) is considered.

Another important characteristic of the QWSC is the fact that it is a soft real-time problem. However, many papers found in the open literature do not explicitly approach this characteristic. Only in some more recent papers this characteristic is mentioned [4] [5]. According to [5], due to this fact, it is necessary to obtain a good solution within the deadline, even if the solution found is only approximate to the optimal one.

E-commerce constitutes an important Internet application that is soft real-time and can benefit from the use of WS [6]. This happens because in complex e-commerce applications, different companies interact and, obviously, they could have different plataforms and languages for their systems. It is also important that e-commerce applications guarantee QoS, avoiding that dissatisfied customers leave the site and do not come back, that would generate monetary losses [7].

An issue related to the QWSC problem and not considered in most of the related papers is the use of statistical techniques to compare different algorithms. In [3] [8] they compared different algorithms using the average response times and the average QoS of them. According to [9] this is not enough and it is necessary to calculate the standard-deviation and confidence interval (in this paper a 95% confidence interval is adopted in all experiments performed).

In this paper, five QoS attributes were defined that are important to e-commerce applications: availability, cost, response time, reputation and confidentiality. Furthermore, five different algorithms were developed to deal with the QWSC problem. A deadline of 1,000 milliseconds to the algorithms was defined and ten different search-space sizes. Also a well-planned performance evaluation experiment was realized, to analyze how it contributes to the optimization of the algorithms for the QWSC problem.

This paper is organized as follows. In Section II the concepts related to WS and QoS are presented. Section III contains the developed algorithms. In Section IV, the testing environment where the experiments were executed is described. The experiment design, which includes fixed and variable factors during experiments, is also presented in this section. In Section V, the results are analyzed according to the response time and QoS obtained. Finally in Section VI, the conclusions are presented and possible future works are discussed.

II. Web Services and QoS

According to the World-Wide Web Consortium (W3C), a WS is defined as: "*A WS is a software system designed to*

 ISBN: 978-1-61208-249-3

support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards" [10]. QoS could be referenced as a set of non-functional properties of Web services, such as performance, reliability, availability and security. With the increasing number of Web services with similar functionality, service quality measures are used to differentiate the existing services [11]. Some of the QoS attributes found in the related works are:

- **Availability**: it is an aspect of quality of service in which a Web service is present or ready for immediate use, represented as a percentage of time available for a service in an observation period and is related to its reliability.
- **Cost**: the amount of money charged by the service provider in order to access the service.
- **Response time**: it is the time spent between the time when the request is made and the time the client receives the response.
- **Reputation**: it is a measure of the client satisfaction by using the service.
- **Confidentiality**: determines that only the receiver and the sender must be able to understand the content of the transmitted message.

Given that each WS has its own QoS attributes, to calculate the QoS of the composition plan as a whole, it is necessary to use aggregate functions [2]. For example, Table I, adapted from [2], shows an example of aggregation of these attributes:

Table I
QUALITY OF SERVICE ATTRIBUTES

QoS Attributes	
Availability	$\prod_{i=1}^{i=n} availability(WSi)$
Cost	$\sum_{i=1}^{i=n} cost(WSi)$
Response Time	$\sum_{i=1}^{i=n} responseTime(WSi)$
Reputation	$\sum_{i=1}^{i=n} reputation(WSi) * 1/n$
Confidentiality	$\sum_{i=1}^{i=n} confidentiality(WSi) * 1/n$

The WS composition plan could be described as a sequence of tasks (abstract WS) with an initial and a final task. For any abstract WS, it could have some candidate services (concrete WS) with same or similar functionality but different QoS attributes. Thus, there are various composition plans for each execution path of composite service. For example, if there is one execution path, with 10 abstract WS and 15 concrete WS per abstract, then the number of composition plans should be about 15^{10} [12]. In [4] it was mentioned that QWSC could be divided into two aspects: QoS-aware selection and orchestration creation. This paper is focused on QoS-aware selection and does not cover the use of Business Process Execution Language (BPEL) for the creation of the execution flow.

III. DEVELOPED ALGORITHMS

This section introduces five different algorithms: Exhaustive Search (ES), Utility Function (UF), Greedy Heuristic (GH), Random Search (RS) and Double Hybrid Genetic Algorithm (DHGA). The procedures of the algorithms and their possible advantages and/or disadvantages will also be discussed in this section.

A. Exhaustive Search (ES)

This algorithm, also known as "brute force", analyses all points in the search space. In the case of the QWSC problem, it compares the QoS obtained by all possible combinations of composite plans and returns the best one (with higher QoS). So the obviously advantage of this algorithm is that the global optima is always guaranteed. The disadvantage is related to their computational complexity, because it is exponential. Suppose a composite flow with ten abstract WS and one hundred concrete WS per abstract WS, the number of points in the search space will be 100^{10} which will probably take hundreds of years to be calculated. Because of that, this algorithm could be used only in small search space sizes, because of the soft real-time characteristic of the QWSC problem.

B. Utility Function (UF)

This algorithm was originally proposed by Yu et. al. (2007) [13] and uses a heuristic utility function to determine the best WS composite plan. It associates each concrete WS a unique QoS value that represents all the QoS attributes of that concrete WS. After that, it selects for each abstract WS the correspondent concrete WS with the higher QoS. Suppose *j* is the current WS to be evaluated, *k* is the number of QoS attributes, μ represents the average value of some QoS attribute, σ the standard-deviation and *q* represents the QoS attribute (i.e. cost, availability, and so on) the Equation 1 represents this algorithm adapted from [13]:

$$Fu(WSj) = \sum_{i=1}^{i=k} \frac{q_i - \mu_i}{\sigma_i} \quad (1)$$

This algorithm has the advantage that it does not need to analyze the entire search space. Another advantage is that its computational complexity is not exponential. For this reason, it could be used in any search space size (in this paper the biggest has 200^{12} points in the search space). The disadvantage of this algorithm is that it could not benefit from a larger deadline. For example, if the deadline was 1,000 milliseconds or one hundred seconds, the QoS obtained would be the same; because this is a deterministic algorithm. It does not guarantee the global optima either.

 ISBN: 978-1-61208-249-3

C. Greedy Heuristic (GH)

This algorithm was an original idea proposed by the authors in [14]. For each abstract WS in the composite flow, the algorithm evaluate all concrete WS available for that abstract WS and selects the one with higher aggregate QoS. Due to all QoS attributes are normalized between 0 and 1 (and the highest is always the best one) it is necessary to calculate the sum of all QoS attributes of all concrete WS. The one with higher aggregate QoS is selected to its respective abstract WS. Suppose *j* is the current WS to be evaluated, *k* is the number of QoS attributes and *q* is the current QoS attribute, the Equation 2 represents the algorithm:

$$GH(WSj) = \sum_{i=1}^{i=k} q_i \quad (2)$$

The advantage of this algorithm is that it is very fast because it is directly related to the number of total concrete WS, i.e., suppose a composite flow with four abstract WS and one hundred concrete WS per abstract WS, the number of total concrete WS will be four hundred. So, the algorithm should calculate the aggregate QoS function of four hundred concrete WS; instead of calculating 100^4 composite plans like the ES algorithm does. The disadvantage of this algorithm is that it could not benefit from a larger deadline, because it is a deterministic algorithm.

D. Random Search (RS)

This algorithm is based on a technique denominated Random Walk. The algorithm randomly moves in the search space, while maintaining the best solution founded and then finishes its execution when the stop criterion is reached (in this case, the deadline). The only advantage of this algorithm is that could benefit from larger deadlines, using all available time to search for a proper solution.

E. Double Hybrid Genetic Algorithm (DHGA)

This algorithm is an original idea proposed by the authors in [14]. It combines a Genetic Algorithm (GA) with the two heuristics (UF and GH) mentioned before. First of all, it runs the UF algorithm and saves its results in a chromosome. After that, it does the same thing for the GH algorithm. So it initializes a random population of chromosomes and then includes the two chromosomes created before in this initial population. The genetic operators used in this algorithm is tournament with 16 players, one-point crossover and elitism operator activated.

The advantages of this algorithm is that it guarantees a solution at least as good as the best of the algorithms UF and GH (because of the elitism operator, that preservers the best solutions through the generations) and the possibilities of obtaining better results when the deadlines increase. The disadvantage is that it cannot guarantee the global optima and it is a slow algorithm for small search spaces.

IV. Performance evaluation

A. Environment Configuration

The main goal of this study is to evaluate different techniques to solve the QWSC problem. Thus, the test environment is composed of three machines: one representing a client, another a service provider and a third one executes a MySQL server with the data about the QoS attributes of the Web services. In the considered environment, the three machines are in the same network and are linked by a gigabit network switch. The machines used are heterogeneous and their configuration is presented in Table II.

The experiments were performed using the default configuration of all tools used. The interaction among the machines is as follows: the client requests a WS composition plan to the service provider, the service provider searches the MySQL server to get data about the QoS attributes of the WS and then executes one of the algorithms and responds to the client who was the WS composition plan selected and its Normalized Composition Aggregated QoS (NCAQ).

B. Experiment design

The experiments were conducted varying three factors in order to verify the performance of the algorithms in the defined deadline and different number of abstract WS and concrete WS per abstract WS. The parameterization of these factors can be observed in Table III.

Another key point to be analyzed in the experiment design is the definition of fixed parameters that should be considered in the application parameterization. Because of the fact that DHGA is a hybridization of a GA and it has to accomplish the established deadline it was defined a population size for all experiments and the number of generations was fixed in five. The populations used for the search-space sizes: 4_100, 4_200, 6_100 and 6_200 was 23,000; for the search-space sizes 12_100 and 12_000 the populations was 21,000.

Table I presents the aggregate functions of QoS attributes considered in this article. However, it is also necessary a way to assess the QoS of the composition as a whole, taking into account the QoS attributes defined. The function to be maximized in the experiments is shown in Equation 3, considering **A** (Availability), **C** (Cost), **RT** (ResponseTime), **R** (Reputation) and **Con** (Confidentiality).

$$F(x) = A + C + RT + R + Con \quad (3)$$

Given that the QoS attributes were normalized in a way that 0 is the worst result and 1 is the best result possible, simply add up all the attributes of QoS, no matter whether they should be minimized or maximized. First, for each QoS attribute, the aggregated QoS is calculated using the formulas presented in Table I. Thereafter, the composition aggregated QoS is computed using the formula shown in Equation 3. Finally, this number is normalized between 0

 ISBN: 978-1-61208-249-3

Table II
ENVIRONMENT CONFIGURATION

Hardware Configuration				
Machine	**CPU**	**Clock**	**Cache**	**RAM**
Service provider	Intel® Core™2 Quad	2.66 GHz	3 MB	8 GB
MySQL server	Intel® Core™i3	3.10 GHz	3 MB	4 GB
Client	Intel® Core™2 Quad	2.4 GHz	4 MB	4 GB

Table III
EXPERIMENTS PARAMETERIZATION

Variable Parameters	
Algorithm	ES, UF, GH, RS and DHGA.
number of abstract WS	2, 4, 6 and 12.
number of concrete WS per abstract WS	100 and 200.
Fixed Parameters	
Deadline	1,000 ms.
Number of replications	10.
Confidence degree	95%

Table IV
GROUP OF EXPERIMENTS

Experiments		
#	**Search-space size**	**Algorithms**
1	2_100	ES, UF and GH.
2	2_200	ES, UF and GH.
3	3_100	ES, UF and GH.
4	3_200	ES, UF and GH.
5	4_100	ES, UF, GH, RS and DHGA.
6	4_200	ES, UF, GH, RS and DHGA.
7	6_100	ES, UF, GH, RS and DHGA.
8	6_200	UF, GH, RS and DHGA.
9	12_100	UF, GH, RS and DHGA.
10	12_200	UF, GH, RS and DHGA.

and 1 and called Normalized Composition Aggregated QoS (NCAQ).

V. RESULT ANALYSIS

In this section, the response times and NCAQ obtained of the five algorithms are analyzed. In Fig. 1 the horizontal axis represents the different search-space sizes evaluated, i.e. "4 - 100" means a composition with four abstract WS and one hundred concrete WS per abstract WS. The vertical axis represents the average NCAQ obtained (the higher the better). It is important to remember that the algorithm called Exhaustive Search (ES) always guarantees the global optima.

A. Time analysis

A deadline of 1,000 milliseconds was defined for the algorithms. The ES algorithm attended this deadline in the experiments groups one, two, three and four. In those cases, the ES was the best choice, because is the only one that guarantees the global optima. In experiments group two and four the three algorithms achieved the global optima, but the only one that can guarantee the global optima for any search-space is the ES. Table V shows the average response times of the experiments. For any search-space size, the three algorithms have accomplished the established deadline.

In the experiments groups five, six, seven, eight, nine and ten, the ES algorithm does not accomplish the deadline. Table VI shows the average response time of these groups of experiments. In experiment group five, the ES algorithm gets really close to the established deadline with an average response time of 1,768 milliseconds. In experiment group six, it takes about 171 seconds (an unacceptable response time) and in experiment group seven, it takes about forty hours, another unacceptable response time for the experiments. With these results, the conclusion is that even for medium search-space sizes, an ES algorithm does not comprise a good choice.

Table V
THE AVERAGE RESPONSE TIME OF ES, UF AND GH IN MILLISECONDS

Search-space size	ES	UF	GH
2_100	421	233	238
2_200	481	238	229
3_100	462	242	225
3_200	488	227	227

B. QoS analysis

Table VII and Fig. 1 show the average NCAQ obtained by the algorithms analyzed. In experiment groups one, two,

 ISBN: 978-1-61208-249-3

Table VI
THE AVERAGE RESPONSE TIME OF UF, GH, RS AND DHGA IN MILLISECONDS

Search-space size	UF	GH	RS	DHGA
4_100	236	234	972	954
4_200	239	240	963	971
6_100	223	240	950	962
6_200	234	250	950	964
12_100	249	233	950	955
12_200	245	253	973	960

three and four the ES algorithm found the global optima within the established deadline, so it could be used in these search-space sizes. In the experiments groups five, six, seven, eight, nine and ten, the ES algorithm does not accomplish the deadline, so it cannot be used, but it was executed to find the global optima and compare it with the other algorithms.

Table VII
THE NCAQ OF ES, UF AND GH.

Algorithm	2_100	2_200	3_100	3_200
ES	0.804	0.846	0.789	0.881
UF	0.801	0.846	0.777	0.881
GH	0.801	0.846	0.777	0.881

In experiment group five the algorithms UF and GH have lower NCAQ than RS and DHGA. Experiment group six shows that UF and GH also have lower NCAQ than RS and DHGA. The RS algorithm has bad results in experiments groups seven, eight, nine and ten.

Finally, in experiments groups seven, eight, nine and ten, the DHGA was always equal or better than the other algorithms (except from ES, but ES could not accomplish the deadline). Because of those results, DHGA was the best algorithm for experiments groups five, six, seven, eight, nine and ten.

C. Influence of factors

The influence of factors was calculated for the algorithms UF and GH. The response variable was the difference between the global optima and the solution obtained by each algorithm. The factors and levels was: **A** - number of abstract WS, **B** - number of concrete WS per abstract WS and **AB** - interaction between A and B. This was done to discover which factors have higher influence on the degradation of the obtained solution. Table VIII shows the influences calculated.

Table VIII
INFLUENCE OF FACTORS

Algorithm	A	B	AB
UF	80%	9%	11%
GH	84%	7%	9%

VI. CONCLUSIONS AND FUTURE WORK

In this paper five different algorithms were presented and evaluated: ES, UF, GH, RS and DHGA. Such algorithms were evaluated considering the response time, the accomplishment of the established deadline and the NCAQ obtained. A testing environment was configured to enable the implementation of the experiments. As mentioned in Section 1, considering a deadline is important because e-commerce applications are soft real-time systems and it is necessary to find good solutions within the established deadline.

Another interesting result shows that is possible to use the design of experiments and performance evaluation to analyze algorithms focused on QWSC. Therefore, future works can consider a more complete performance evaluation including more GA operators for the DHGA algorithm, such as: crossover operator, mutation operator, mutation rate, crossover rate and so on.

In future works it is planned to analyze the influence of factors: number of abstract WS in the composition flow and number of concrete WS per abstract. By doing that, it could be possible to determine which algorithm is the best one for each search-space size; i.e. for a “2 - 100” search-space size the ES algorithm is probably the best choice, because it guarantees the global optima and comprises the deadline; for a “4 - 100” search-space size DHGA is the best choice. Combining the information of the influence of the deadline and the influence of the search-space size, it could be possible to dynamically determine which algorithm is the best (or at least, a good choice) for each requisition to a WS Composition Engine, that will execute the appropriate QWSC algorithm.

It is also planned in the next steps to use the design of experiments and performance evaluation to develop new hybrid algorithms that should probably have a better performance than some of the GAs present in the related works found in the literature.

REFERENCES

[1] G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani, “An approach for qos-aware service composition based on genetic algorithms,” in *Proceedings of the 2005 conference on Genetic and evolutionary computation*. New York, NY, USA: ACM, 2005, pp. 1069–1075.

[2] J. M. Ko, C. O. Kim, and I.-H. Kwon, “Quality-of-service oriented web service composition algorithm and planning architecture,” *Journal of Systems and Software*, vol. 81, no. 11, pp. 2079 – 2090, 2008.

[3] B. Batouche, Y. Naudet, and F. Guinand, “Semantic web services composition optimized by multi-objective evolutionary algorithms,” in *Internet and Web Applications and Services (ICIW), 2010 Fifth International Conference on*, may 2010, pp. 180 –185.

 ISBN: 978-1-61208-249-3

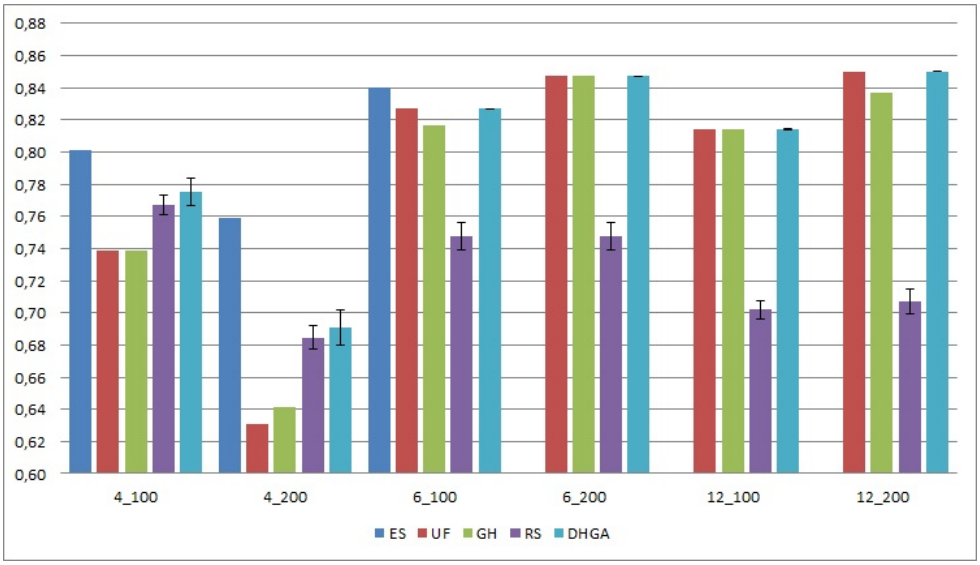

Figure 1. The average NCAQ of all algorithms.

[4] Y.-Y. Fanjiang, Y. Syu, C.-H. Wu, J.-Y. Kuo, and S.-P. Ma, "Genetic algorithm for qos-aware dynamic web services composition," in *Machine Learning and Cybernetics (ICMLC), 2010 International Conference on*, vol. 6, july 2010, pp. 3246 –3251.

[5] H. Liu, F. Zhong, B. Ouyang, and J. Wu, "An approach for qos-aware web service composition based on improved genetic algorithm," in *Web Information Systems and Mining (WISM), 2010 International Conference on*, vol. 1, oct. 2010, pp. 123 –128.

[6] M. Bravetti, R. Lucchi, G. Zavattaro, and R. Gorrieri, "Web services for e-commerce: guaranteeing security access and quality of service," in *Proceedings of the 2004 ACM symposium on Applied computing*. New York, NY, USA: ACM, 2004, pp. 800–806.

[7] D. A. Menascé, D. Barbará, and R. Dodge, "Preserving qos of e-commerce sites through self-tuning: a performance model approach," in *Proceedings of the 3rd ACM conference on Electronic Commerce*, ser. EC '01. New York, NY, USA: ACM, 2001, pp. 224–234.

[8] L. Ai, M. Tang, and C. Fidge, "Partitioning composite web services for decentralized execution using a genetic algorithm," *Future Generation Computer Systems*, vol. 27, no. 2, pp. 157 – 172, 2011.

[9] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*, 1st ed. Wiley, apr 1991.

[10] W3C, "Web services architecture," 2012, available at: http://www.w3.org/TR/ws-arch/. Last access: 02/26/2012.

[11] S. Kalepu, S. Krishnaswamy, and S. Loke, "Verity: a qos metric for selecting web services and providers," in *Web Information Systems Engineering Workshops, 2003. Proceedings. Fourth International Conference on*, dec. 2003, pp. 131 – 139.

[12] C. Zhang and Y. Ma, "Dynamic genetic algorithm for search in web service compositions based on global qos evaluations," in *Scalable Computing and Communications; Eighth International Conference on Embedded Computing, 2009. SCALCOM-EMBEDDEDCOM'09. International Conference on*, sept. 2009, pp. 644 –649.

[13] T. Yu, Y. Zhang, and K.-J. Lin, "Efficient algorithms for web services selection with end-to-end qos constraints," *ACM Trans. Web*, vol. 1, no. 1, May 2007.

[14] P. F. do Prado, L. H. V. Nakamura, J. Estrella, M. J. Santana, and R. H. C. Santana, "Different approaches for qos-aware web services composition focused on e-commerce systems," in *XIII Simposio em Sistemas Computacionais, 2012 WSCAD-SSC*, october 2012, pp. 179 – 186.

 ISBN: 978-1-61208-249-3