Server and Path Selection in a Light Architecture Content Streaming System with Dual Adaptation

Eugen Borcoci, Marius Vochin, Mihai Constantinescu,
University POLITEHNICA of Bucharest
Bucharest, Romania
emails: eugen.borcoci@elcom.pub.ro,
marius.vochin@elcom.pub.ro,
mihai.constantinescu@elcom.pub.ro

Jordi Mongay Batalla, Warsaw University of Technology / National Institute of Telecommunications
Warsaw, Poland
jordim@interfree.it
Daniel Negru, LaBRI Lab, University of Bordeaux,
Bordeaux, France
daniel.negru@labri.fr

Abstract — **Media content streaming and delivery is nowadays a high popularity service in Internet. Complex architectures like Content Delivery Networks, Content Aware/Oriented Networks have been proposed, where information/content objects are treated as first-class abstractions. As alternative, light architectures, are investigated and implemented, working on top of the current IP networking technologies. In both types of architectures, appropriate content server and path selection constitute the primary set of actions to be performed in the content delivery systems. Such a problem belongs to the general class of multi-criteria optimization problem, having (in our case) as input, some information on servers, network and user context. This paper contains an extension of a preliminary work, focused on algorithms and policies for optimized paths and server selection. Simulation study results are presented here, to illustrate the better performance of multi-criteria optimization algorithms versus random path and/or server selection. Flexibility of the solution is emphasized, including the possibility to naturally add new criteria (business, policies) in the selection process. This work is a part of a larger effort, aiming to finally implement a subsystem in the framework of a content delivery light architecture system.**

Keywords — Content delivery; Server selection; Path selection; Content-Aware Networking; Multi-criteria decision algorithms; Dual adaptation; Future Internet.

I. INTRODUCTION

The content orientation is an important trend recognized in the current and Future Internet [1][2]. Consequently, several solutions have been recently proposed, studied and implemented, aiming to better support the content oriented services. The *Information/Content-Centric Networking* (ICN/CCN) approach [3][4] revisits some main concepts of the architectural TCP/IP stack; the novelty is that in ICN/CCN the information/content objects are treated as first-class abstractions. In such architectures the intelligence and complexity of the network nodes are higher; the routers can perform content-based route computation (routing) and forwarding, caching and other content-oriented processing, leading to systems better adapted to the content requests and delivery, in comparison with traditional network. However, the cost of such systems is rather high, both due to the architectural changes and also due to the much higher processing performance required from routers. Therefore, some more “light ICN” evolutionary solutions, preserves the main TCP/IP concepts, but introduce a degree of *Content-Awareness at Network* layer (CAN) [5]. Seen partially as an orthogonal solution, *Content Delivery Networks* (CDNs) improve the content services [6] by distributing the content replica to cache servers, located close to groups of users. However, all the above solutions involve complex management and control architectures, high CAPEX and significant modifications to be introduced by Service/Content Providers and Network Providers/Operators.

As alternative, over-the-top (OTT) solutions are investigated, where the high level services are delivered over the connectivity offered in current Internet. Here, a Service Provider (SP) is not (fully) responsible for the transmission of the information flows to the end-user; users access is done via the “public Internet”. An OTT-type SP could be an entity separate from the traditional Internet Service Provider (ISP). Also, combined solutions exist, with OTT Service Providers using the CDN Providers infrastructure to improve the quality of delivery.

A light architecture (OTT-like), for content streaming systems over the current Internet is proposed by the European *DISEDAN* Chist-Era project [7][8], (*service and user-based* ***DI****stributed* ***SE****lection of content streaming source and* ***D****ual* ***A****daptatio****N*****,** 2014-2015). The business actors involved are: *Service Provider (SP),* which is an entity/actor that delivers the content services to the users and possibly owns and manages the transportation network); *End Users (EU),* which consumes the content; a *Content Provider* could exist, which is the owner of some *Content Servers*. However, DISEDAN does not deal with contractual relationships between the CP and SP; one may therefore assume, in a simplified model, that severs are also owned by the SP.

A novel concept is introduced based on:

(1) a *two-step server selection mechanism* (at SP and at EU) using algorithms that consider context- and content-awareness. An effective solution is constructed for the multi-criteria hard problem of best content source (server) selection, considering user context, servers’ availability and requested content.

(2) a *dual adaptation mechanism* consisting of Media adaptation (also called *media flow adaptation*) and content source adaptation (by *switching the streaming server*), when the quality observed by the user suffers degradation during the media session.

The proposed solution could be rapidly deployed in the market since it does not require complex architecture like CON/ICN, full-CAN or CDN.

This paper is an extension of the work ([1], ICSNC 2014 conference paper) on paths and server combined selection algorithms and policies applicable by SPs in a system having light content delivery architecture. This contribution additionally presents and analyzes a lot of experimental results. The acquisition of the input information for the selection procedure is out of scope of this work; it is supposed that such information is provided statically or dynamically (by measurements performed by a monitoring subsystem) and made available for the algorithm.

Section II is a short overview of related work, focused on multi-criteria optimization algorithms and their adaptation to our context. Section III describes at high level the overall system and outlines the server-path selection problem. Sections IV and V contain the main paper contributions, focused on paths and content server selection combined algorithm, simulation models and results. Section VI proposes modifications of the MCDA algorithms to allow introduction of SP policies, aiming to increase the system flexibility. Section VII contains conclusions and future work outline.

II. RELATED WORK

A. *Multi-criteria decision Algorithms*

This section is a short overview on some previous work related to *path-server selection* in content delivery systems, based on *Multi-Criteria Decision Algorithms (MCDA)*. The problem belongs to the more general one, known as *multi-objective optimization*. This has been extensively studied in various and large contexts of economics and engineering. The paper will not detail this. Few references are given at the end of the paper [9][10].

The general problem of multi-objective optimization is to minimize $\{f_1(x), f_2(x), \ldots, f_m(x)\}$, where $x \in S$ (set of feasible solutions), $S \subset R^n$.

A *decision vector* is defined as $x = (x_1, x_2, \ldots, x_n)^T$.

One might have $(m \geq 2)$ possibly conflicting *objective functions* $f_i : R^n \rightarrow R$, i= 1, ..m and *we would want to minimize them simultaneously.*

A set of *Objective vectors* is defined as images of decision vectors, where *objective (function) values* are :

$$z = f(x) = (f_1(x), f_2(x), \ldots, f_m(x))^T$$

It is called a *feasible objective region* W = f(S) the image of S in the objective space.

Objective vectors are said to be optimal if none of their components can be improved without deterioration to at least one of the other components.

A decision vector $x\_ \in S$ is defined as Pareto optimal if there does not exist another $x \in S$ such that $f_i(x) \leqslant f_i(x\_)$ for all $i = 1, \ldots, k$ and $f_j(x) < f_j(x\_)$ for at least one index j.

Figure 1 shows an example for Pareto front, where x = (server, path); n= 2, $x \in Z^2$ (the paths and servers are identified through some positive integer indexes). We have $f(x) = (f_1, f_2)$ = (srv_load, 1/ path_avail_ bandwidth), m= 2.

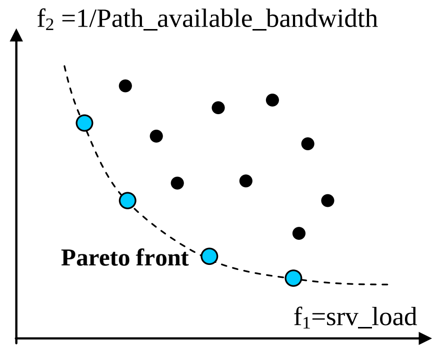


Figure 1. An example of objective space and Pareto front

Such optimization problems are in general NP complete, so, different simplified heuristics have been searched. A simple scalar approach maps the k-dimensional vector onto a single scalar value *w* by using an appropriate cost function c(), thus reducing the problem to a single-criterion one. However, information about individual components is lost. In the server-path selection problem several decision parameters are important, such as: server load and proximity, transport path (length, bandwidth, loss, and jitter).

Solutions have been searched, treating the decision variables separately and considering them as independent. Note that in our case this is only partially true, e.g., delay and jitter are clearly not independent variables. Therefore, modifications should be added to the basic algorithm to capture such effects and this paper proposes a solution.

In the following, we will use a simplified notation:

- identify the solutions directly by their images in the objectives space R^m ; in other words, we define as decision parameters/variable the set v_i , i = 1, ..m, with $\forall i, v_i \geqslant 0$, where they are actually the values of the objective functions;

- S is number of candidate solutions; they are indexed by s = 1, 2, ..S;

- the image of a candidate solution s is $Sl_s=(v_{s1},v_{s2},...,v_{sm})$ represented by a point in R^m.

Value ranges of decision variables may be bounded by given constrains. The selection process means to select a solution satisfying a given objective function conforming a particular metric.

One approach to solve the optimization problem is *reference level decision algorithms* [11-13]. This will be used in the paper. Considering the above notations, the basic algorithm defines two reference parameters:

- r_i =*reservation level*=the upper limit for a decision variable which should not be crossed by the selected solution;

- a_i=*aspiration level*=the lower bound for a decision variable, beyond which the solutions are seen as similar.

Without loss of generality one may apply the definitions of [14][15], where for each decision variable v_i there are defined r_i *and* a_i, by computing among all solutions s = 1, 2, ..S:

$$r_i = \max [v_{is}],\ s = 1, 2, ..S$$
$$a_i = \min [v_{is}],\ s = 1, 2, ..S \qquad (1)$$

Figure 2 shows a simple example, where m= 2.

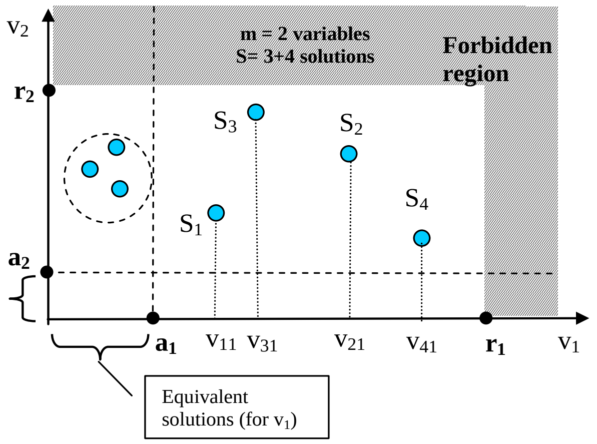


Figure 2. An example of reference level limits (aspiration and reservation)

In [13], modifications of the decision variables are proposed: *replace each variable* with *distance from it to the reservation level*: $v_i \rightarrow r_i - v_i$ (so, increasing v_i will decrease the distance); normalization is also introduced *to get non-dimensional values, which can be numerically compared*. For each variable v_{si}, a ratio is computed, for each solution *s*, and each variable i:

$$v_{si}' = (r_i - v_{si})/(r_i - a_i) \qquad (2)$$

The factor $1/(r_i-a_i)$ - plays also the role of a weight. The variable having high dispersion of values *(max – min)* will have lower weights, and therefore greater chances to determine the minimum in the next relation (3). In other words, less preference is given to those variables having close values (reason: a given candidate selection among them will not influence significantly the overall optimum).

The algorithm steps are:

Step 0. Compute the matrix M$\{v_{si}'\}$, *s*=1…S, i=1…m

Step 1. Compute for each candidate solution *s*, the minimum among all its normalized variables v_{si}':

$$min_s = min\{v_{si}'\};\ i=1...m \qquad (3)$$

Step 2. Make selection among solutions by computing:

$$v_{opt} = max\, \{min_s\},\ s=1, ..S \qquad (4)$$

This v_{opt} is the optimum solution, i.e it selects the best value among those produced by the Step 1.

The reference level algorithm has been used in several studies.

The work [15] proposes a decision process for network-aware applications, based on reference level MCDA with several metrics. The improvement (compared to the basic algorithm) consists in considering not only the currently selected server status, but also the system future state after the selection. The simulation results showed a slight gain versus the basic algorithm, while using the same information from the network level (server and link load).

The work [16] proposes and evaluates a multi-criteria decision algorithm for efficient content delivery applicable to CDN and/or ICN. It computes the *best available source and path* based on information on content transfer requirements, servers and users' location, servers load and available paths. It runs processes at two levels: 1. *offline* discovering multiple paths, and gathering their transfer characteristics; 2. for each content (online) request, finding the best combined server – path (reference level model). The following "use cases" are evaluated: *random server and random path*, combined with shortest single path routing protocol (current Internet solution); *closest server and random path*, (similar to the current CDN); *least loaded server and random path*; *best server and the path with more available bandwidth* in the bottleneck link. Simulation, using Internet large scale network model, confirmed the effectiveness gain of a content network architectures (i.e., having a degree of network awareness) and efficiency of the combined path-server selection.

The work [17] models and analyzes a simple paradigm for *client-side server selection*. Each user independently measures the performance of a set of candidate servers, randomly chooses two or more candidate and selects the server providing the best hit-rate. The algorithm converges quickly to an optimal state where all users receive the best hit-rate (respectively, bit rate), with high probability. It is also shown that if each user chooses just one random server instead of two, some users receive a hit-rate (respectively, bit rate) that tends to zero. Simulations have evaluated the performance with varying choices of parameters, system load, and content popularity.

The contributions of this paper w.r.t. previous work mentioned are summarized as: two-phase flexible selection procedure based on MCDA reference level algorithm, applicable with slight modifications for nine use cases (see Section IV); additional policy supporting modifications proposed for the basic algorithm, in order to capture different Service Provider strategies.

Note that other algorithms can be used to optimize the selection belonging to a different class like *Evolutionary Multi-objective Optimization Algorithms (EMO)* [18]. However, this approach is not in the scope of this study.

B. Dual Adaptation

The adaptation techniques of interest for DISEDAN have been : in–session media flow adaptation methods (*Dynamic Adaptive Streaming over HTTP (DASH)* [19-20]) and complemented by the *Content Servers* (CS) switching (this is partially similar to the first CS selection).

The DISEDAN DASH subsystem details are not in the scope of this paper. However, for the sake of completeness we gave a short description of it. This technology has been selected in DISEDAN for in-session media adaptation.

DASH was recently adopted as multimedia streaming standard, to deliver high quality multimedia content over the Internet, by using conventional HTTP Web servers [19-20]. It minimizes server processing power and is video codec agnostic; it enables automatic switching of quality levels according to network conditions, user requirements, and expectations. The DASH offers important advantages (over traditional push-based streaming) [21]. A DASH client continuously selects the highest possible video representation quality that ensures smooth play-out, in the current downloading conditions. This selection is performed on-the-fly, during video play-out, from a pre-defined discrete set of available video rates and with a pre-defined granularity (according to video segmentation).

III. DISEDAN SYSTEM SUMMARY

A. System Concept

The DISEDAN project proposes an *evolutionary and light architecture* for content delivery via Internet, multi-domain compatible. It works in Over the Top (OTT) style, involving more simple management and control in comparison to ICN/CCN. DISEDAN defines a novel concept having as main features, as mentioned in Introduction section: a. *two-step server selection mechanism* (at Service Provider (SP) and at End User) by using algorithms that consider context- and content-awareness; b. *dual adaptation mechanism during the sessions*, - which consists in media flow adaptation (based on segmented video content delivery by using *Dynamic Adaptive Streaming over HTTP (DASH)* [19-20]) and/or content servers switching (handover).

Note that DASH details and design are out of scope of this paper.

Figure 1 illustrates the system concept. The main business entities/ actors are those mentioned above: SP, EU, CS. The SP and possible Content Provider (CP) entities are not seen as distinct. Also, the full *CS* management is out of scope of this system. The connectivity between CSs and EU Terminals (EUT) are assured by traditional *Internet Services Providers (ISP) / Network Providers (NP)* - operators. The ISP/NPs do not enter explicitly neither in the business relationships set considered by DISEDAN, nor in the management architecture.

It is supposed that SP has knowledge on: Content servers identities, their status - including their availability in terms of content objects contained and their current available capacity to serve new requests. Also, the SP may (optionally) have some information about static and/or dynamic connectivity resources (at overlay level) in the network, between different servers and potential groups of users (the latter could be placed everywhere in the network). Such information - if it exists - comes from external entities (e.g, network operators) or even from the CSs, if they are capable to evaluate the characteristics of some paths.

A simplified scenario is described below (see Figure 3).

The End User issues a *Media description request* (action (1) in Figure 3) to SP. The SP analyses the requests; it evaluates the CSs status and after applying an optimization algorithms, returns to the user a *Media Program Description (MPD) file*– containing, among others, the identity of a selected server (or a ordered list of servers). The End User performs the final selection based on SP-delivered information and - possibly - based also on some local measurements. Again, an optimization algorithm could be used in this final selection. The End User selects Content Source 1 and starts to ask (HTTP requests) video segments, (2).

While receiving the segments, local quality measurements (3) are performed at End User Terminal (EUT) in order to guide the adaptation process. If, for some reasons, the received quality was poor, then the EUT may decide:

- either to apply DASH adaptation, i.e., to determine the rate for the next segment request and maybe ask the next segment with a lower bit rate (5a) from the same Content source, or
- to switch to another CS. In this case, it may initiate probing of other candidates (4) and finally switch the Content source (in our case Content Source 2 is selected as a new source).

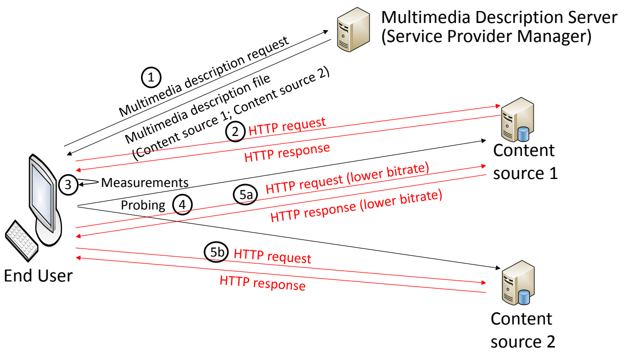

Multimedia Description Server
(Service Provider Manager)
1
Multimedia description request
Multimedia description file
(Content source 1; Content source 2)
2
HTTP request
HTTP response
3
Measurements
Probing
4
5a
HTTP request (lower bitrate)
HTTP response (lower bitrate)
5b
HTTP request
HTTP response
End User
Content source 1
Content source 2

Figure 3. DISEDAN concept

B. System Architecture

Figure 4 shows a simplified high level view of the general architecture.

The Service Provider includes in its Control Plane:

- *MPD File generator* – dynamically generates Media Presentation Description (MPD) XML file, containing media segments information (video resolution, bit rates, etc.), ranked list of recommended CSs and, optionally - current CSs state information and network state (if applicable).
- ***Selection algorithm*** – runs Step 1 of server selection process. It exploits *Multi-Criteria Decision Algorithms (MCDA)* [9][15][16], modified to be applied to DISEDAN context, or *Evolutionary Multi-objective Optimization algorithm (EMO)* [18], etc.,

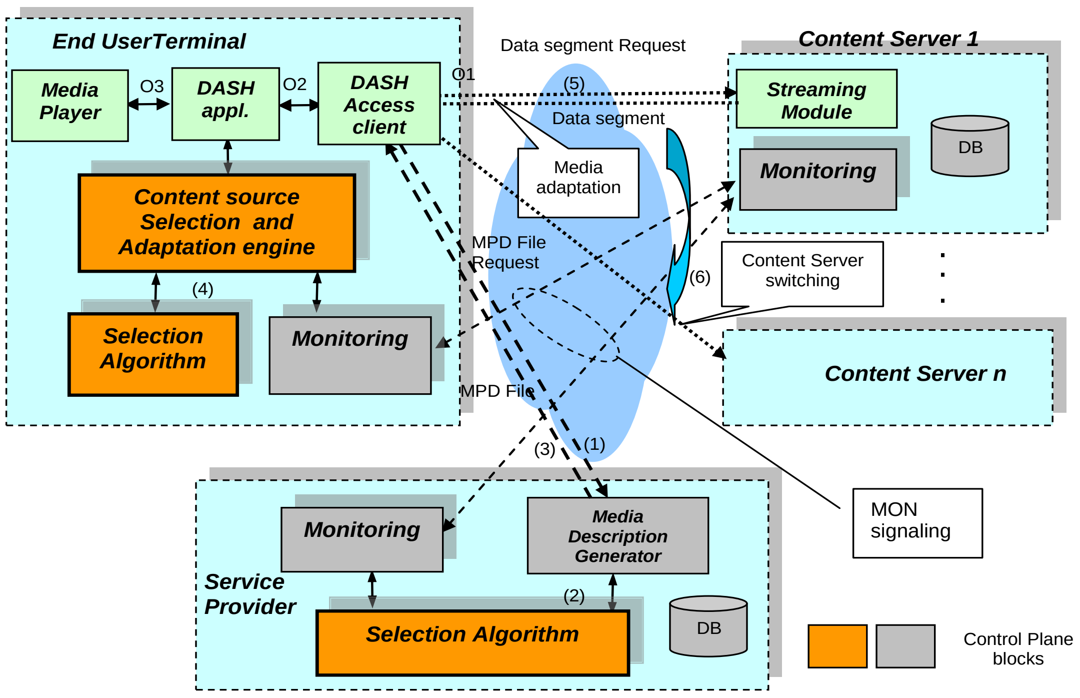


Figure 4. DISEDAN general architecture; DASH - Dynamic Adaptive Streaming over HTTP; MD – Media Description; DB – Data Base ; O1, O2, O3 – DASH Observation Points [ISO/IEC 23009-1]

to rank recommended CSs and media representations, aiming to optimize servers load as well as to maximize system utilization.

- *Monitoring module* – collects monitoring information from CSs and performs the processing required to estimate the current state of each CS.

The End User Terminal entity includes the modules:

- *Data Plane: DASH (access and application)* – parses the MD file received from SP and handles the download of media segments from CS; *Media Player* – playbacks the downloaded media segments.
- *Control Plane:* ***Content Source Selection and Adaptation engine*** – implements the dual adaptation mechanism; *Selection algorithm* – performs the Step 2 of server selection process. It can also exploit MCDA, EMO, or other algorithms to select the best CS from the set of candidates recommended by SP; It runs adaptation process; *Monitoring module* – monitors changing (local) network and server conditions.

The Content Server entity includes the modules:

- *Data Plane: Streaming module* – sends media segments requested by End Users;
- *Control Plane*: *Monitoring module* – monitors CS performance metrics (CPU utilization, network interfaces utilization, etc.).

The following functional steps are performed:

(1) EUT issues to SP a media file request.

(2) SP analyzes the status of the CSs and runs the selection algorithm (optionally the SP could make first, a current probing of the CSs); For each user request the SP could consider also the user profile, the policies of the SP for this user's class and other information at the SP side (e.g., states of the servers and possibly network-related information).

(3) SP returns to EUT a ordered list of candidates CS (SP proposal) embedded in a MD- xml) file.

(4) The EUT performs the final CS selection, by running its own selection algorithm.

(5) EUT starts asking segments from the selected CS. During media session, the EUT makes quality and context measurements. Continuous media flow adaptation is applied using DASH technology if necessary or (6) CS switching is decided. From the EU point of view the steps 1-2-3 composed the so-called Phase 1 and steps 4-5-6 the Phase 2.

During the receipt of consecutive chunks, the user's application can automatically change the rate of the content stream (internal DASH actions, - which are out of scope in this paper) and/or also can switch to another CS.

IV. PATH AND SERVER SELECTION OPTIMIZATION ALGORITHM

A two phase selection process is adopted here, similar to [9][15]. The Phase 1 is executed offline and computes candidate paths from servers to users. The Phase 2 applies a MCDA (reference level variant) algorithm and computes the best path-sever solution, based on multi-criteria and also policies guidelines. Note that the multi-criteria algorithm is flexible: any number of decision variables can be used, depending on their availability. For instance, in a multi-domain network environment it is possible that SP has not relevant or complete knowledge about end to end (E2E) transport paths. In such cases the list of available decision variables can be as well used. Another additional contribution here consists in modifying the reference algorithm, to include different SP policies concerning the importance of some decision variables with respect to others.

A. Network Environment

The content delivery might involve several network domains independently managed [4][5]. In a combined optimization procedure for path and server selection it is not realistic, from the real systems management point of view, to consider all details of the paths from the content servers to the users. Therefore (supposed in this paper and also in DISEDAN), the SP network awareness is limited, e.g., to knowledge about the inter-domain context, i.e., the inter-domain graph (where each network domain is abstracted as a node) and inter-domain link capacities, while considering the multi-tier organized Internet. The location (domains) of the potential groups of users and server clusters are also supposed to be known.

Figure 5 shows a generic example of a tiered structure network, containing several domains D11, ...D33 interconnected via inter-domain links. At the edges of this structure, groups of servers and users are connected to Tier 3 domains. In Figure 5, two possible paths from D33 to D32 are shown. The Phase 1 procedure will compute such similar paths between two edge domains.

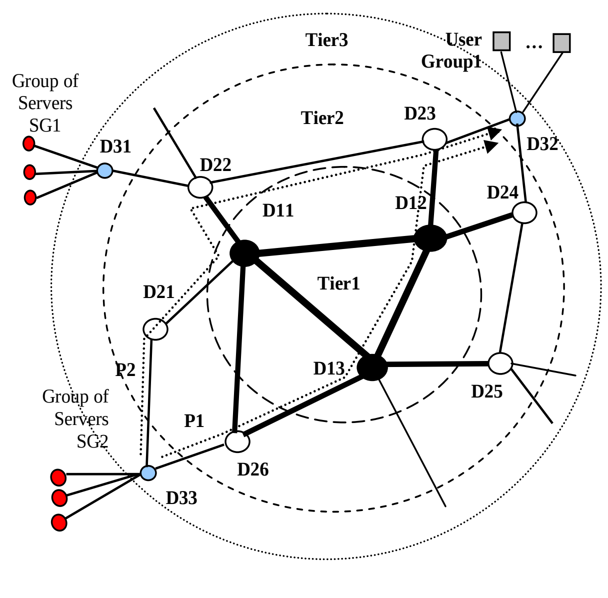


Figure 5. Example of a sample tiered network. P1 and P2 – paths from domain D33 to D32.

B. Use cases for path-server selection procedure based on MCDA algorithm

Several Use Cases can be defined for a combined algorithm, by considering several criteria for the path and server selection. Different metrics can be defined for paths and servers status evaluation. The path metric can be the simplest - number of hops, or a more powerful one (enabling better QoS assurance), e.g.: link cost=1/B, where B could be the static link capacity or the available bandwidth (dynamically measured). Also, constrained routing policies can be applied (e.g., related to bandwidth, number of hops, etc.). The bandwidth of the selected path should be the maximum one (among several paths having the same start and end nodes) but evaluated at the bottleneck link of that path. Additionally, the path might be constrained, e.g.: the number of hops (i.e., domains), should be lower than a maximum. For server status, one could consider the server proximity to the user, or server load. The MCDA algorithm has the quality that it can use several decision variables and make a global optimization.

For the path selection one may apply: a. *Single path between server and user* (usually provided by the current Internet routing based on minimum number of hops); b. *Random path selected among equal costs paths* between server and user, given that a multipath protocol is applied (e.g., modified Dijkstra algorithm); c. *best path among several paths* having similar costs in a defined range.

For server selection one may apply: 1. *Random selection*; 2. *Closest server* to the user (e.g., considering as metric the number of hops, i.e., domains - between server and user; 3. *Least loaded server* (the load can be evaluated as the current number of connections, or partially equivalent- as the total bandwidth consumed at the server output).

Considering combinations of the above factors, nine Use cases (and corresponding algorithms) can be defined: a.1, a.2, ...c.3, if independent decisions are taken for path, and respectively the server, with no MCDA algorithm. However, we will consider a global optimization MCDA algorithm with several decision variables taken from the above.

C. Two phases path-server selection procedure

The following simplifying assumptions are considered valid for this first version of the selection procedure:

- All servers are managed by the unique module called *Resource Allocator (RA)* belonging to SP Manager. The RA knows each server status, including its current load (number of active connections and bandwidth consumed at the server output). A degree of content-awareness exists in RA; it knows the inter-domain graph, and inter-domain link capacities.
- Each domain is considered as a node in the network graph, i.e., the intra-domain transport is not visible. This is a major realistic assumption in simplifying the amount of knowledge supposed to exist at SP level.
- All servers and users location are established offline, and are fixed. However, the system can accommodate the end user terminal mobility, given that in the content delivery phase a content server switching is possible.

- The total number of content objects (*Max_no_CO)* are distributed (offline mode, by an external caching process, out of scope of this algorithm) to server groups and between the servers of a given group, while the number of COs in a server should be ≤ *Max_no_CO_per_Server*.
- The content object instances replicated in surrogate servers are known by the RA. A data structure *CO_SRV _map* contains the mapping of CO replica on servers. Each CO is stored in 1, 2…. K servers; K = maximum number of servers to replicate a content object.
- The time-life of a CO instance in a server is unlimited.
- All COs are delivered in unicast mode, so a "connection" is 1-to-1 mapped to a content consumption session. The COs have the same popularity.
- Each CO user request asks for a single CO; however, the same CO can be consumed simultaneously by several users, by using private connections.
- RA treats the User requests in FIFO (queue named *COreq_Q*) order.
- RA accepts or rejects user requests. Rejection happens if there are no servers, or no transport resources available. No further negotiation between the User and RA is assumed after a request transaction processing.
- The bandwidth occupied by a connection is equal to *Bw_CO* (in the first approach it can be considered constant). More generally this bandwidth is random, in a range Bw_CO +/- ΔBw.
- A connection load for the server and path will be Bw_CO, during Tcon interval measured from the connection request arrival instant (we neglect the processing time for content/connection requests).
- RA uses the most simple additive bandwidth management (no statistical multiplexing is assumed).
- The average duration of a connection (for content consuming) is Tcon. The real duration could be in a range TCon +/- ΔTcon.

The Phase 1 (offline) general objective is to compute, on the inter-domain graph, (multiple) paths from server domains to user domains. No traffic load consideration is applied. The input data are: topology, inter-domain link capacities, location of servers, and users. Some constraints can be applied, e.g., bottleneck bandwidth (BB) on any path ≥ Bmin; number of hops (domains) on any path ≤ NHmax. The simplest metric is the classic one (number of hops).

More powerful approaches compute multiple paths: equal cost paths, or sets of paths having costs in a given range. Having more than one path would provide several MCDA choices opportunity. The multiple paths can be computed, by running a modified version of the classic Djikstra algorithm [22]. A "better" (from QoS point of view) additive metric is: *link_cost= 1/*B_{link}, where B_{link} is the link bandwidth/capacity). Given that routing process is a classic one, it will be not detailed in this paper. The Phase 1 output is a set of sub-graphs, each one containing the multi-paths from a given group of servers to a given group of users. The Phase 1 algorithm is convergent. Its order of complexity is not higher than for different variants of Dijkstra based algorithms [22].

Phase 2

The Phase 2 of decision process jointly selects (for each user request arrived at RA), the best pair server-path (based on dynamic conditions) from the available candidates computed in the Phase 1. The signalling details user-RA are out of scope of this paper. The RA applies an admission control decision, followed/combined with an MCDA algorithm. The Phase 2 dynamicity means updating the paths and server loads according to the new requests arrived. Also considering the time-life of a connection, different server status items are updated when the connections are terminated. Note that there is no problem to downgrade the algorithm if complete path information is missing. More generally, the number of decision variables and the amount of information existent on them (static and/or dynamic) are flexible items in MCDA approach.

A description of Phase 2 is given below in a free-style simplified pseudo-code format.

Description of Phase 2 in pseudo-code

Request analysis and resource allocation (pseudocode)

// It is assumed a time process, which triggers activation of the main procedure, at each generic time tick instant Tk. This approach can serve also for managing the time lives of connections. The algorithm description is given below.

```
 Each Tk
{   While COreq_Q non-empty
        {req = Extract_first_element_from COreq_Q( );
         Process_request (req);//processes the first request from the request queue COreq_Q
        Adjust_time_life_of_connections_in servers; }
 }

Process_request(req) // description of a user request processing
        {Identify_Server _groups_and_individual_servers_able_to_provide_CO ; // candidate servers for requested content
                {//Search in the CO_SRV _map, by using the CO index in the request}
        Create _candidate_servers_vector; // containing one entry for each such server
        Collect_status_of_each_server;  //from a data structure Server_status, the status of each sever is loaded in the
```

```
                                              // vector; in the most simple variant the status is: the current number of active
                                              //   connections for that server
        Determines_ sub-list_of _paths_for each candidate_server; // from the list of updated paths, by using information
                                                                  // from the Phase 1
        Create_candidate_list_of_path_server_solutions; //each solution is characterized by server load, bandwidth
                                                       // and number of hops
        Delete_full_loaded_servers; //optional; it can be included in MCDA algorithm
        Delete_elements_ from_the_list_of_paths_associated_to_the_candidate_list:// optional; it can be done by MCDA
               // those which have number of hops > NHopmax
               // those which have Available Bandwidth < Bmin

        Run_the_MCDA reference_level_ algorithm ;// determine best path-server solution; policies can be included here

        If successful
            then
              {Increase_success_list_statistics;
               Update_the_allocated_server_load;
                   // Increase the number of active connections
                   // Load & start timer associated to the time-life of this connection
               Add_additional_bandwidth_consumed_to_ the_allocated_path_load on all links;}
            else increase_the_reject_list_statistics;
        }
Adjust_time_life_of_connections // delete the terminated connections from the server status
        For each server //Sv1, …Svn
           { For each timer
               { If Active_flag=1 and Timer_value >0
                     then {Timer_value - -;
                           If Timer_value = 0 then {Active_flag=0; NCO_srv --;}}}
Generation_ Request_for_Content_object_
        Initialization: TReq = random [1,…P*Tk];
        Each Tk // equivalent with periodic interrupts at Tk seconds interval
               {Treq = Treq - 1;
                  If Treq =0   then
                      { k = random [1, …. Max_no_CO];
                       Put_CO_req (User_id, Tcon, COk,)_in_COreq_Q;//place a new request in the queue
                      TReq = random [1,…P*Tk]}; // restart timer and select a random interval until the
                                                // next request generation}
```

V. SIMULATION SCENARIOS AND SAMPLES OF RESULTS

Considering the assumptions presented in the previous section and optimization algorithm, several simulation scenarios have been elaborated and experimented. This section presents a summary of results obtained. The objective is to determine the degree of effectiveness that an algorithm as MCDA can have versus other more simple selection decisions.

A. *Simulation Framework*

Figure 6 shows the multi-domain network model used in simulations. The network topology is organized on three tiers: Domains 5 and 6 belong to Tier1; Domains 3, 4, 8 and 9 belong to Tier2; Domains 1, 2, 7, and 10 belong to Tier3.

The general approach of the simulation framework is described below.

A *two phase selection* process is adopted. The prerequisites are: topology, EUs, servers are fixed; content object distribution is random but static;

Phase 1 executed offline, computes candidate paths from servers to users. *Phase 2 executed* online, applies a selection algorithm (MCDA, etc.) and allocates resources; statistics are collected.

The EU requests are addressed to RA and are randomly distributed (uniform) in time. The RA searches a solution: runs a selection procedure (MCDA, etc.) to assign a (server, path) pair for this request; Updates the system status (servers, paths); counts the *success* and *reject* events in variable traffic load conditions

In order to compare MCDA results to other selection procedures the simulation model also supports:

Best server (BS) (least loaded)- single criterion :

1. get *Request (obj_id)* from an EU
2. select the set of servers having *obj_id*

3. select among them a BS *(no matter the path)*
4. Final decision: fail if *no_avail_ Srv* or *no_avail_path*, *accept* otherwise
5. If, *accept*, then update the system status

Closest server (CS) single criterion :
1., 2., same as before
3. Select CS *(no matter the path)*
4., *5*. Same as before

Scalar metric(CO1) (Minkowski- order 1)
1., 2., same as before
3. Compute metric, select the solution having M(1) = min
4., *5*. Same as before

The Minkowski metric of order 1 [16] is computed as :

Cost = NH/NHmax + SL/SLmax + PL/PLmax, where NH = number of hops on a path, SL= server load, PL= path load.

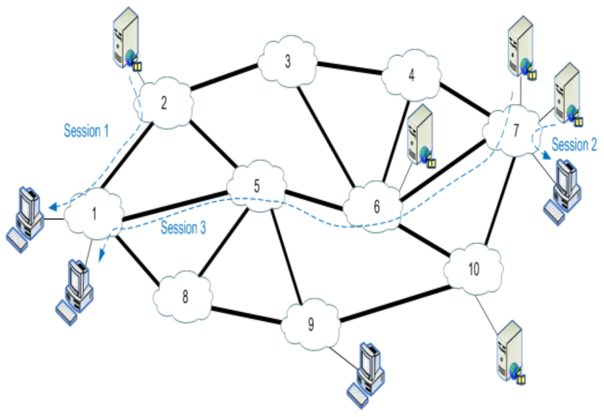


Figure 6. Network model used in simulations

B. Scenario 1 Results

Sample of the simulation results are presented below. The simulation model parameters are:

int Smax = 50; //Max no. of servers deployed over the whole network

int Cmax = 100; //Max no. of original content objects

int D = 10; //The number of network domains within the whole network

int Nmax = 300; //Max no. of user requests within the simulation time interval

int T = 1000; //The simulation time interval

int M = 300; //The lifetime for a server-user multimedia streaming session

int CD = 15; //Max number of content object replica within whole network

There are several streaming servers on each domain. The users are requesting for different multimedia contents. For any accepted request, a new server-user session is opened.

Figures (7-10) show comparative results for different algorithms: the success ratio versus request rate. One has considered different *replication factors*, here samples are shown for versus *rf*= 3, 8 replicas. The following comments on results can be stated:

- When the system is significantly non-saturated (low request rate) different algorithms produce similar results. No significant gain is observed for MCDA.
- When the system saturation is very high, again the algorithms results start to be similar. Therefore, for a given network dimensioning one should evaluate the region (versus traffic load), in which more complex algorithms such as MCDA may produce significantly better results, versus trivial ones.
- When replication factor *rf* increases the MCDA results are clearly better. One can observe an improvement ratio of 40% in case of *rf* = 8.
- When replication factor *rf* increases the Minkowski-1 metric-based algorithm produces similar results as MCDA.

Note: The following diagrams (in Scenarios 1 and 2 of results) have the “request rate” label on oX axis. Actually, the values are the total request number per simulation time. However, the label was set with attribute “rate” to emphasize the fact that the request rate is increasing proportionally as the total number of request is higher.

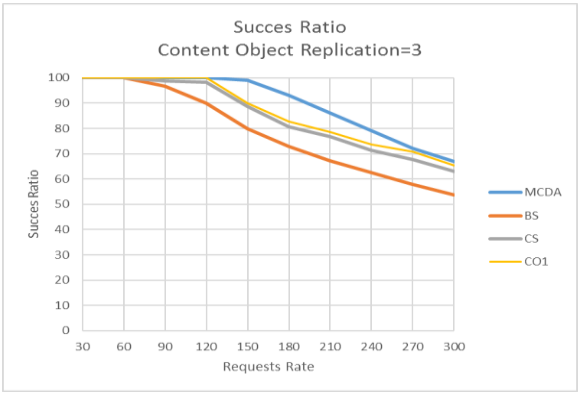


Figure 7. Success ratio for CO replication = 3

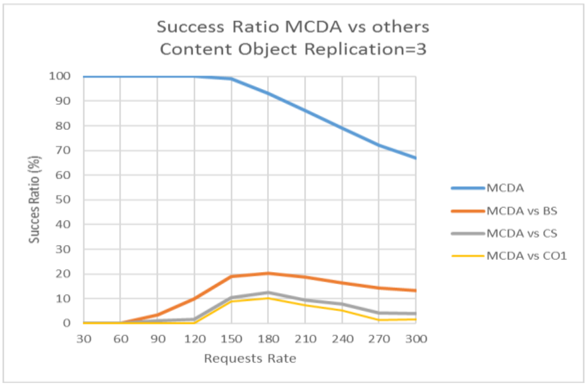


Figure 8. MCDA gain for CO replication = 3

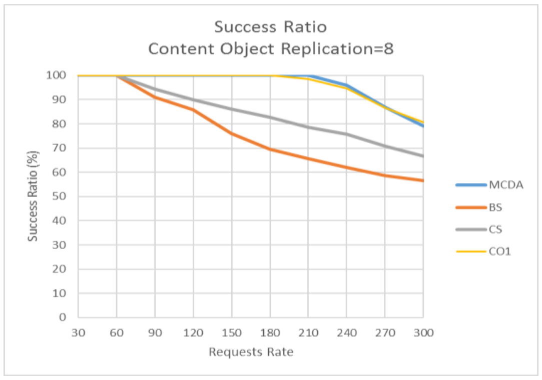


Figure 9. Success ratio for CO replication = 8

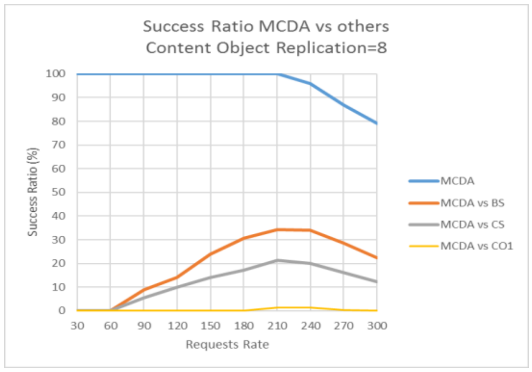


Figure 10. MCDA gain for CO replication = 8

Figure 11 presents a summary of experiments showing how MCDA success ratio varies when different replication factor is applied and different request rates exist. The surface is called "decision space", because one can decide based on such surfaces when it is worth to apply a MCDA algorithm, and when other more simple approach could be used.

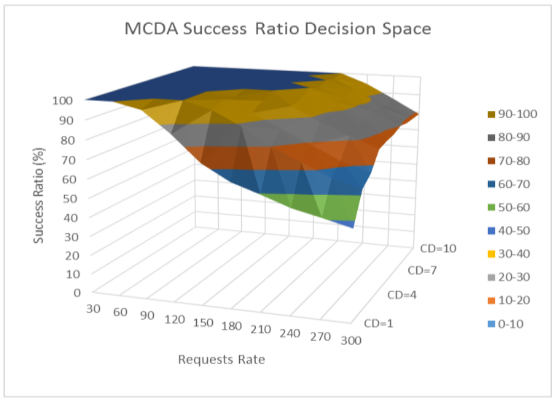


Figure 11. Summary of MCDA success ratio versus request rate and replication factor

Figure 12 serves to illustrate more the previous statement associated to Figure 11; it shows the MCDA improvement versus Best Server selection. It is again seen that MCDA can offer significant gain only if the replication factor is higher than 5 and for higher request rates.

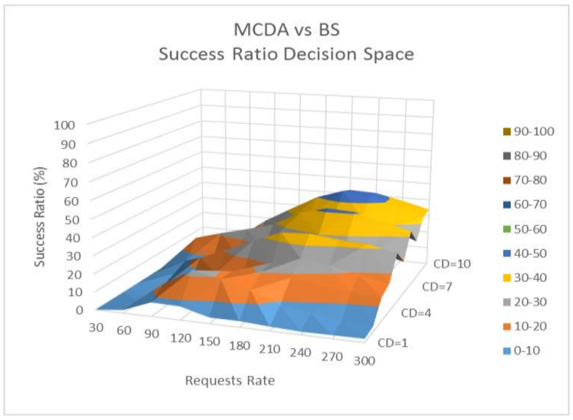


Figure 12. Summary of MCDA success ratio versus request rate and replication factor

C. *Scenario 2 Results*

The same network has been used. Some specific conditions of these experiments have been:

Simulation Interval T=100

Max number of requests Nmax=variable = [100…5000];

Rate = Nmax/T

Multimedia session duration M=30

Link capacities: tier1=100, tier2=30, tier3=10

Number of replicas for the content objects:4.

Note: The diagrams below have been extracted from experiments where it is wanted to check the correctness of rejection reasons due to different non-available resources (either server saturation or link/network paths saturation). In order to emphasize such effects, an increasing additional load in the network has been generated. Instead of adding background traffic the method to increase the network load has been by not releasing in the network the paths occupied by a session after its termination (however, the server is freed). In this way, the network traffic is constantly increasing during the simulation time. Consequently, it is expected that a correct behavior of the algorithm will determine more rejections due to network links saturation.

Figure 13 shows in the above condition the performance of the Best Server method selection, given that the total traffic in the network is increasing (two reasons: request rate increasing and, the percentage of rejections due to path load is increasing). This shows that in real cases the network connectivity capacities should be enough high, otherwise a high number of servers does not help so much.

A similar behavior is exposed when the selection is performed based on Minkowski - order 1 metric- see Figure 14. Figure 15 shows that still MCDA has ~20% gain over Best-server method even at high request rate. However, the gain is less between MCDA and CO-1 (Figure 16). The explanation of this is that CO-1 includes several parameters in its scalar formula.

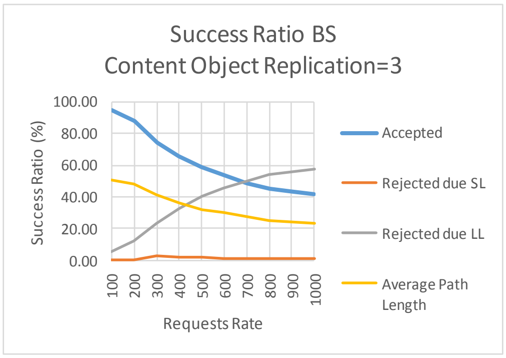


Figure 13. Best Server (BS)- performance

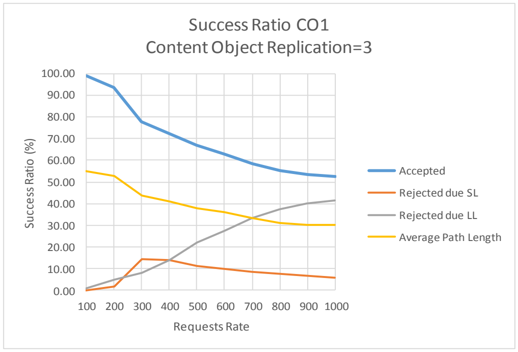


Figure 14. CO-1 - performance

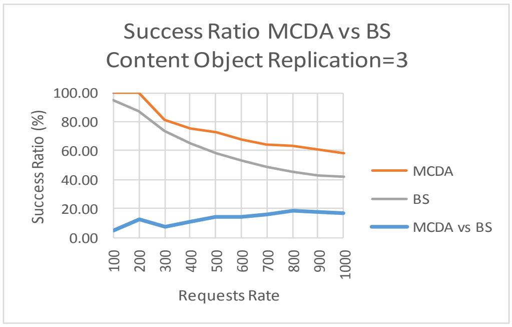


Figure 15. MCDA gain versus BS

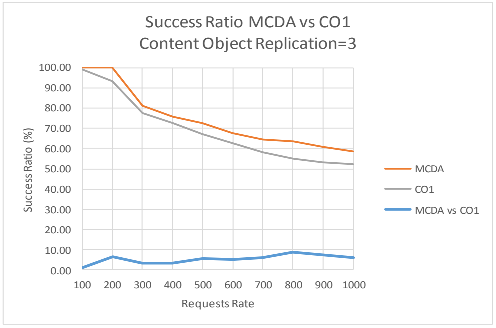


Figure 16. MCDA gain versus CO-1

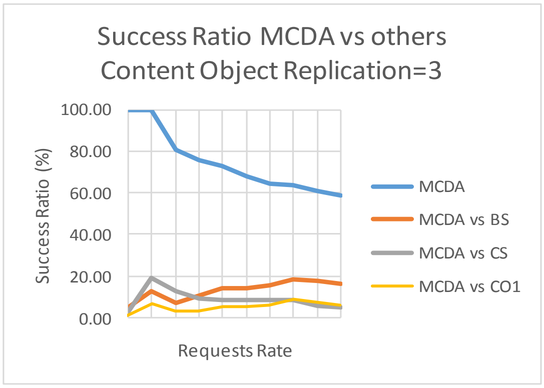
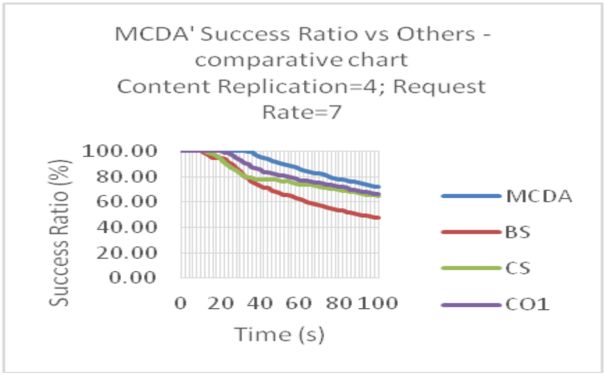
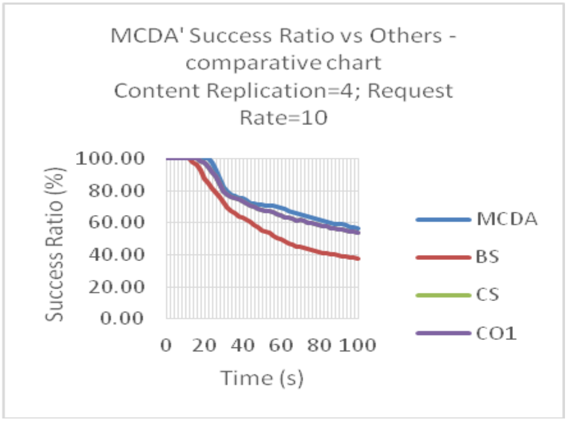


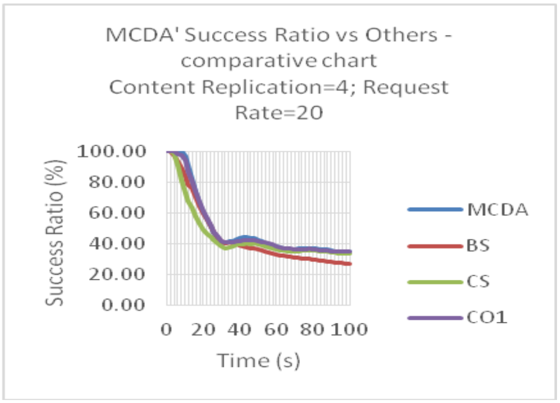
Figure 17. MCDA gain versus others

Figure 17 shows an aggregated diagram: MCDA gain versus other methods. It is seen the closest to MCDA is CO-1.

Figure 18 contains four diagrams representing the behavior of the system versus time. One can see that for a high request rate the system becomes saturated sooner and the MCDA does no more offer gain versus other selection methods (e.g. rate = 7 versus rate = 20, or 50)







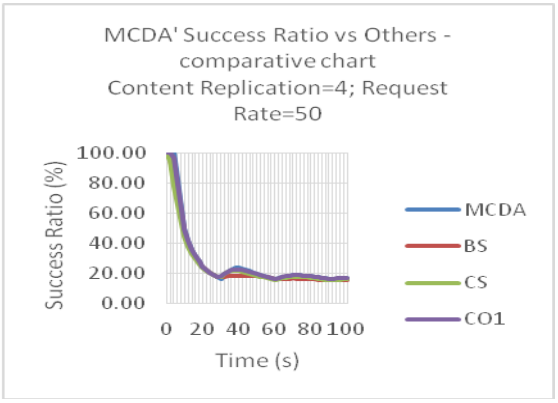
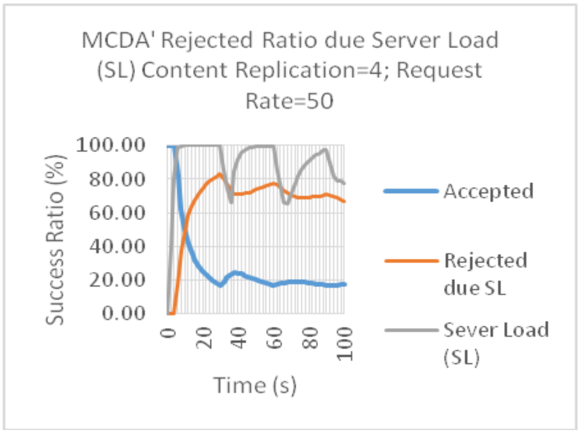


Figure 18. System behavior versus time for different request rates, going towards system saturation

At high request rate and rather equal session duration (T≈ 30 sec.) a cyclic behavior is observed (Figure 19). When the system is unloaded, a lot of requests are accepted in short time. Then the servers become saturated (100% load). The sessions terminate at t ≈ 30 sec., all rather in the same short interval (compared to session duration) and Server load decreases sharply. Meantime new requests arrive (however, the network is partially loaded now) and the Server Load increases again until t ≈ 60 sec, when a new decrease is observed and so on.



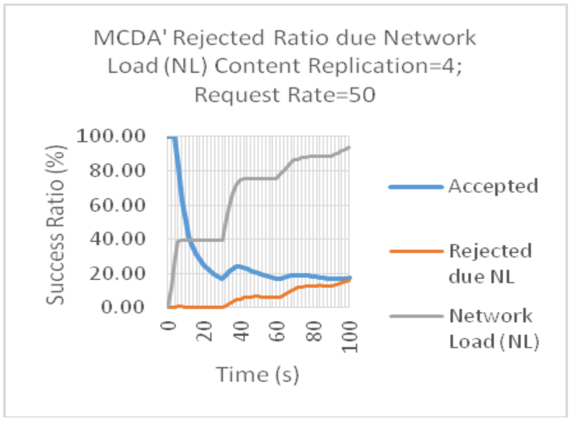
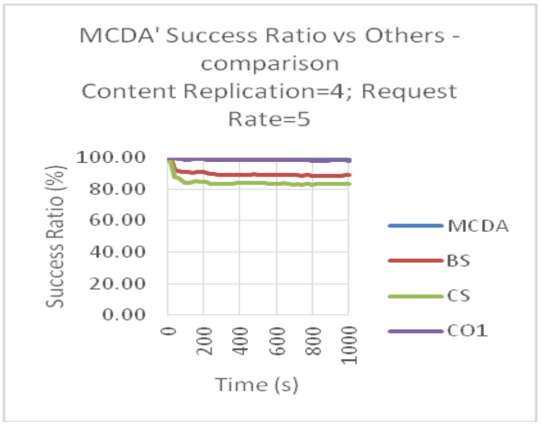


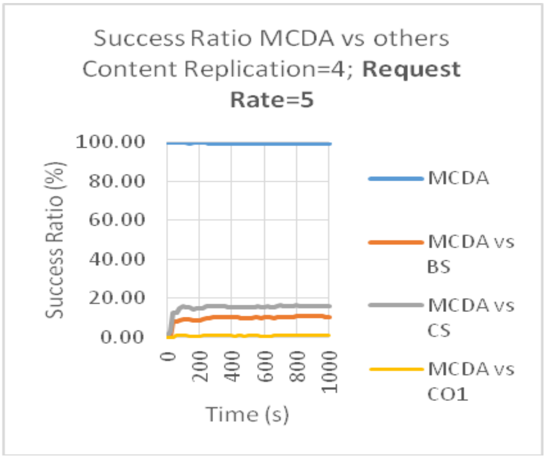
Figure 19. MCDA behaviour at high request rate

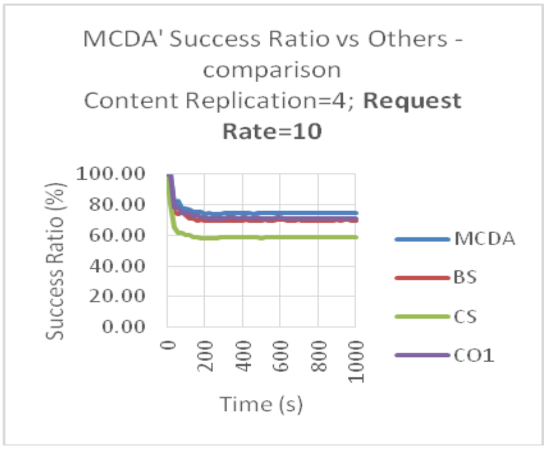
D. *Scenario 3 Results*

The conditions of simulations are the same as in Scenario 2 except the fact that no other additional traffic is present in the network but only the media session traffic. Each session when terminated will determine release of the resources on all network links previously used.

The simulation time has been increased to allow the time diagrams to show the region of some stable conditions of load for servers and network paths.







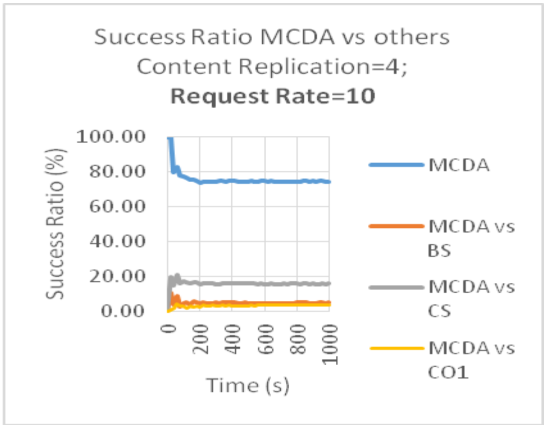


Figure 20. Comparison of MCDA behavior versus other algorithms versus time

The results presented in Figure 20 show a better performance of MCDA evaluated on long term, versus other methods. The MCDA algorithm is compared to Best Server, Closest Server and Minkowski 1- metric algorithm. The main conclusion is that MCDA and CO1 expose very close performances while BS and CS are weaker- due to either server or path non availability.

VI. POLICY GUIDING THE MCDA

Several remarks can be done related to the basic reference level algorithm:

- The formula *min_s=min{ v_{si}' }; i= 1..m (3)*, selects as *representative of each candidate solution*, the “worst case” value, i.e., for all other variables/parameters, this solution has “better” normalized values then this representative. This is arithmetically correct, however, in practice, this “worst” case parameter might be actually less important than others, either from technical or business (i.e., policies) point of view.
- In some particular cases with dependent variables (e.g., delay/jitter) the solution selected could be not the most appropriate, from actual implementation point of view.
- The step 2 *compares values coming from different types of parameters* (e.g., 1/Bwdth, delay, jitter, server load, etc.) - independent or dependent on each other. The normalization allows them to be compared in the *max{ }* formula. However, the numbers compared are from items having different nature. *This is an inherent weak property of the basic algorithm.*
- More important is that the SP might want to apply some policies when selecting the path-server pair for a given user. Some decision variables could be more important than others. For instance, the number of crossed domains (no_of_hops in MCDA) can be the most important parameter – given the transit cost. In other cases, the server load could be more important, etc.

A simple modification of the algorithm can support a variety of SP policies. We propose here a modified formula:

$$v_{si}' = w_i(r_i-v_{si})/(r_i-a_i) \qquad (3')$$

where the factor $w_i \in (0,1]$ represents a weight (priority) that can be established from SP policy considerations, and can significantly influence the final path-server selection. This will solve the above mentioned issues.

A sample example below shows the optimization obtained. Let us consider a selection scenario, in which the decision variables are given in Table I, and six candidates in Table II (entries are native not-yet normalized values)

Priorities are introduced in Table I, derived from SP policy. Here, the server load and numbers of hops are considered the most important.

One can define: a1= 0, r1=100; a2=0, r2=10; a3=110, r3 = 10; a4=0, r4=50; a5=0, r5=100.

TABLE I. DECISION VARIABLES EXAMPLE

Decision variables	Semantics	Units	Priority
v1	server load	(%)	1- max
v2	number of hops	Integer	1
v3	available bwdth on the path	Mbps	2
v4	jitter	ms	3
v5	E2E delay	ms	4- min

TABLE II. CANDIDATE SOLUTIONS EXAMPLE

	s_1	s_2	s_3	s_4	s_5	s_6
v_{s1}	0	20	40	70	80	100
v_{s2}	5	7	6	3	4	5
v_{s3}	40	20	50	80	50	60
v_{s4}	0	10	30	20	10	30
v_{s5}	30	80	70	40	30	50

Applying the basic algorithm (i.e., with no priorities) simple computation will show that formula (4) is *max{0.3, 0.1, 0.3, 0.3, 0.2, 0}*, showing that solutions s1, s3, s4 are equivalent. However, examining the initial input candidate values, it is clear that s_1 is the best (server load=0, and sufficient available bandwidth- compared to others).

Now, we introduce policies, assuming the priorities assigned in Table I. Some weights (acting as compression factors) can be defined, e.g., $w_1= 0.5, w_2= 0.5, w_3= 0.7, w_4= 0.8, w_5= 1.0$. Then applying the formula (3'), one gets a new set of values for the formula in (4), i.e., *max {0.21, 0.07, 0.2, 0.15, 0.1, 0}*. It is seen that s1 solution is now selected as the best, which corresponds to the intuitive selection of it.

Some other examples have been checked to verify the prioritized selection capability of the modified MCDA. Note that despite its simplicity the modification proposed can have major impact on algorithm results, given that different SP policies can be defined, depending on user categories, content server exploitation needs, networking environment, etc. Therefore, the weighting factors in practice do not come from some formulas, but should be chosen, based on the defined priorities of the SP. A natural usage of the modified algorithm proposed here could be to select several sets of best solutions, fit to the different policies of the Service Provider.

VII. CONCLUSIONS AND FUTURE WORK

This paper presented a study on multi-criteria decision algorithms and procedures for best path-server selection in a content delivery system.

While applying some previous ideas of two phases procedure (offline and online) the solution adopted here is a flexible (supporting many use cases) modified decision procedure, which additionally can capture some policy related priorities for decision variables. It was shown that such modifications can enhance the added value of the decision taken by the algorithm.

Future work will be done (in the DISEDAN project effort) to simulate the system in a larger network environment, and finally, to implement the described procedures in the framework of a system dedicated to content delivery based on a light architecture.

ACKNOWLEDGMENTS

This work has been partially supported by the Research Project DISEDAN, No.3-CHIST-ERA C3N, 2014- 2015 and project POSDRU/107/1.5/S/76903.

REFERENCES

[1] E. Borcoci, M. Vochin, M. Constantinescu, J. M. Batalla, and D. Negru "On Server and Path Selection Algorithms and Policies in a light Content-Aware Networking Architecture," ICSNC 2014: The Ninth International Conference on Systems and Networks Communications http://www.thinkmind.org/index.php?view=article&articleid=icsnc_2014_1_40_20077

[2] http://www.chistera.eu/sites/chistera.eu/files/DISEDAN%20-%202014.pdf, retrieved: 07, 2014

[3] J. Choi, J. Han, E. Cho, T. Kwon, and Y. Choi, "A Survey on Content-Oriented Networking for Efficient Content Delivery," IEEE Communications Magazine, March 2011, pp. 121-127.

[4] V. Jacobson et al., "Networking Named Content," CoNEXT '09, New York, NY, 2009, pp. 1–12.

[5] J. Pan, S. Paul, and R. Jain, "A survey of the research on future internet architectures," IEEE Communications Magazine, vol. 49, no. 7, July 2011, pp. 26-36.

[6] P. A. Khan and B. Rajkumar. "A Taxonomy and Survey of Content Delivery Networks". Department of Computer Science and Software Engineering, University of Melbourne. Australia : s.n., 2008. www.cloudbus.org/reports/CDN-Taxonomy.pdf.

[7] FP7 ICT project, "MediA Ecosystem Deployment Through Ubiquitous Content-Aware Network Environments," ALICANTE, No. 248652, http://www.ict-alicante.eu, Sept. 2013.

[8] http://wp2.tele.pw.edu.pl/disedan/, retrieved: 07, 2014.

[9] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering". Struct Multidisc Optim Eds., No. 26, 2004, pp. 369–395.

[10] J. Figueira, S. Greco, and M. Ehrgott, "Multiple Criteria Decision Analysis: state of the art surveys," Kluwer Academic Publishers, 2005

[11] J. R. Figueira, A. Liefooghe, E-G. Talbi, and A. P. Wierzbicki "A Parallel Multiple Reference Point Approach for Multi-objective Optimization," "European Journal of Operational Research 205, 2 (2010), pp. 390-400, DOI: 10.1016/j.ejor.2009.12.027.

[12] A. P. Wierzbicki, "The use of reference objectives in multiobjective optimization". Lecture Notes in Economics and Mathematical Systems, vol. 177. Springer-Verlag, pp. 468–486.

[13] T. Kreglewski, J. Granat, and A. Wierzbicki, "A Dynamic Interactive Decision Analysis and Support System for Multicriteria Analysis of Nonlinear Models," CP-91-010, IIASA, Laxenburg, Austria, 1991, pp 378-381.

[14] J. M. Batalla, A. Bęben, and Y. Chen, "Optimization of the decision process in Network and Server-aware algorithms", NETWORKS 2012, October 15–18 2012, Rome.

[15] A. Beben, J. M. Batalla, W. Chai, and J. Sliwinski, "Multi-criteria decision algorithms for efficient content delivery in content networks," Annals of Telecommunications - annales des telecommunications, vol. 68, Issue 3, 2013, pp. 153-165, Springer.

[16] E. Borcoci, ed., et al., D2.1 "System requirements and comparative analysis of existing solutions for media content server selection and media adaptation," July 2014, http://wp2.tele.pw.edu.pl/disedan/

[17] C. Liuy, R. K. Sitaramanyz, and D. Towsleyy, "Go-With-The-Winner: Client-Side Server Selection for Content Delivery," http://arxiv.org/abs/1401.0209, retrieved: 07, 2014.

[18] J. Mongay Batalla, C. X. Mavromoustakis, G. Mastorakis, D. Négru, and E. Borcoci, "Evolutionary Multiobjective Optimization algorithm for two-phase content source selection process in Content Aware Networks," submitted to Springer 4OR - A Quarterly Journal of Operations Research.

[19] I. Sodagar, "The MPEG-DASH Standard for Multimedia Streaming Over the Internet," MultiMedia, IEEE, vol. 18, no. 4, pp. 62 - 67, 2011.

[20] ETSI TS 126 247 V11.7.0 (2014-07) (UMTS); LTE; Transparent end-to-end Packet-switched Streaming Service (PSS); Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH), (3GPP TS 26.247 version 11.7.0 Release 11, 2014).

[21] O. Oyman and S. Singh, "Quality of Experience for HTTP Adaptive Streaming Services," IEEE Communications Magazine, April 2012, pp.20-27.

[22] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, "Introduction to Algorithms," The MIT Press, Cambridge, Massachusetts, 2000, ISBN: 0-262-53091-0.