On Server and Path Selection Algorithms and Policies in a light Content-Aware Networking Architecture

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 —**Appropriate content server and path selection procedures based on different algorithms constitute the first set of actions to be performed in content delivery systems. Multi-criteria optimization algorithms based on user context, network and servers information can be used to enhance the overall efficiency. This paper contains a preliminary work, focused on algorithms and policies for optimized paths and server selection, 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, Future Internet.

I. INTRODUCTION

The content orientation is an important trend recognized in the current and Future Internet [1] The Information/Content-Centric Networking (ICN/CCN), approach [2][3], revisits some main concepts of the architectural TCP/IP stack. In parallel, "light ICN". evolutionary solutions introduce Content-Awareness at Network layer (CAN) [4]. Seen partially as an orthogonal solution, Content Delivery Networks (CDNs) improve the content services [5] by distributing the content replica to cache servers located close to groups of users. However, the above solutions involve complex architectures, high CAPEX and significant modifications in Service/Content Providers and Network Providers/Operators.

The DISEDAN Chist-Era project [6][7], (service and user-based **DI**stributed **SE**lection of content streaming source and **D**ual **A**daptatio**N,** 2014-2015) proposes an *evolutionary and light architecture* to enhance the content delivery via Internet. It studies pragmatic solutions for the multi-criteria hard problem of best content source selection, considering user context, servers availability and possibly network status information (if available). The novel concept is based on: 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*, consisting of media flow adaptation and/or content servers handover. The solution could be rapidly deployed in the market since it does not require complex architecture like ICN, full-CAN or CDN.

This paper contains a preliminary work on paths and server combined selection algorithms and policies applicable by SPs in a light content delivery architecture Section II is a short overview of related work. Section III outlines the overall system and problem description. Section IV contains the main paper contributions, focused on: a. paths and content server selection combined algorithm; b. modifications to allow introduction of SP policies, aiming to increase the system flexibility. Section V contains conclusions and future work outline.

II. RELATED WORK ON MULTI-CRITERIA DECISION ALGORITHMS

This section is a very 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 [8][9][12].

The general problem of multi-objective optimization is to find $min\ F(x) = [f_1(x), ..f_k(x)]$ where $x \in X^i$, the decision variables space, and $f_1(x), ..f_k(x)$, are a set of objectives, [8] [9]. Such 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.

The *reference level decision algorithms*, [10][11], considers a decision space R^m and the decision parameter/variables: v_i, i=1, ..m; $\forall i$, $v_i \geq 0$. A candidate solution is an element $S_s=(v_{s1}, v_{s2}, .., v_{sm}) \in R^m$. Let S be the number of candidates indexed by s = 1, 2, ..S. The value ranges of decision variables might be bounded by given constrains. The selection process searches a solution satisfying a given objective function, conforming a particular metric.

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 [11], where for each decision variable v_i there are defined r_i *and* a_i , by computing among all solutions s = 1, 2, ..S:

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

In [11], modifications of the decision variables are proposed: *replace each variable* with *distance from it to the reservation level*: $v_i \rightarrow r_i\text{-}v_i$; (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:

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

The factor $1/(r_i\text{-}a_i)$ - plays also the role of a weight. The variable having high dispersion of values (max – min) will have lower weights, and so, 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: 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 [13] 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 [14] 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 [15] 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.

III. DISEDAN SYSTEM SUMMARY

The DISEDAN solution performs an initial path-server selection and then, during the session, applies media flow adaptation based Dynamic Adaptive Streaming over HTTP (DASH) and/or content server handover. Details are described in [6], [7]. The system has a light architecture in the sense that it does not mandatory assume special Management and Control Plane at SP and end user sides. However the SP can provide to the client, at least a list of available and appropriate servers and/or other (offline and/or online observed) information to optimize the final selection at EU side selection results The design is backwards-compatible: both (un)modified client and/or SP can cooperate. Based on the evaluated current delivery conditions, rules are defined to decide which adaptation action to perform. The DISEDAN implementation will be flexible [6], [7], allowing cheap and seamless deployment.

This paper is focused on the path-server selection problem, applicable to DISEDAN. 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) and made available for the algorithm.

IV. PATH AND SERVER SELECTION OPTIMIZATION

A two phase selection process is adopted here, similar to [14]. 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 multicriteria 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.

i. Network Environment

The content delivery for large communities of users frequently 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 network awareness of the management and control entity of an SP 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 1 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 1, two possible paths from D33 to D32 are shown. The Phase 1 procedure will compute such similar paths between two edge domains.

ii. 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. Several 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 link should be the maximum one (among several paths) 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).

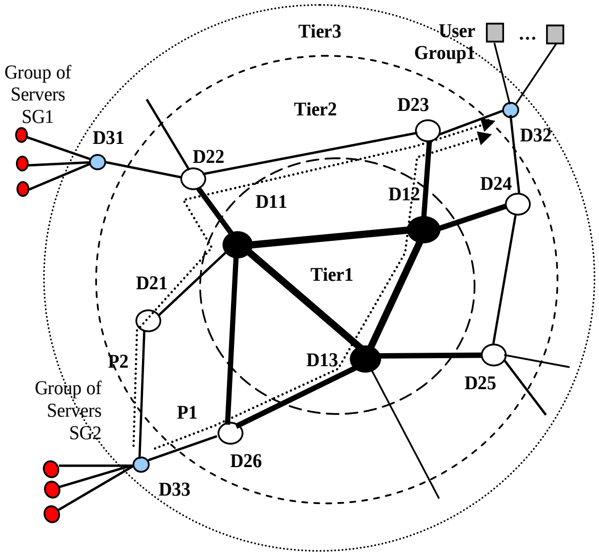

Figure 1. Example of a sample tiered network. P1 and P2 – paths from domain D33 to D32.

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.

iii. 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 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.

Phase 1

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 [16]. 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, [17].

Phase 2

The Phase 2 of decision process jointly selects (for each user request arrived at RA), the best server and 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.

A description of Phase 2 is given below.

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 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 current number of active connections
        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_Content_object_request_
     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;//generate content object request
               TReq = random [1,…P*Tk]}; // restart timer and select a random interval until the
                                         // next request generation}
```

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 1, and six candidates in Table 2 (entries are native not-yet normalized values)

Priorities are introduced in Table 1, 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 1. 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.

V. CONCLUSIONS AND FUTURE WORK

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

[2] 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.

[3] V. Jacobson, et al., “Networking Named Content,” CoNEXT ’09, New York, NY, 2009, pp. 1–12.

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

[5] 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.

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

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

[8] 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.

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

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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

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