Server Replication in Multicast Networks

Hamed S. Kia and Samee Ullah Khan
Department of Electrical and Computer Engineering
North Dakota State University
Fargo, ND, 58108–5060, USA
{hamed.sajjadikia, samee.khan}@ndsu.edu

***Abstract*—This paper studies and proposes heuristic algorithms to solve the problem of replicated server placement (RSP) with Quality of Service (QoS) constraints. Although there has been much work on RSP in multicast networks, in most of them a simplified replication model is used; therefore, their proposed sultions may not be applicable to real systems. In this paper we use a more realistic, and generalized model for replica placement, which considers the latency restriction of the receivers (QoS), bandwidth, and storage constraints of the links and nodes. We present a mathematical formulation and propose four heuristics that are benched marked using BRITE network generator, and discuss the benefits and drawbacks of the static and dynamic approaches. The proposed heuristics are experimentally compared through simulations with respect to their performance and computational complexity under different QoS constraints. The simulation results show interesting charecteristics of the studied heuristics.**

I. Introduction

Due to the increased number of Internet broadcast applications, efficient multicast protocols have become more important than ever [1]- [3]. Server replication has been shown to be one of the most effective mechanisms to cope with multicast network reliability. A replicated server services requests within a close neighborhood. Thereby, if properly designed, then such local servicing can significantly reduce network traffic.

In this paper, compared to previous works [1]- [3], we consider a generalized replicated server placement (RSP) problem, which considers the latency restriction of the receivers, bandwidth, and memory constraints of the links and nodes. We must notice that the proposed placement algorithms are static in nature. However nothing precludes them from being classified as semistatic placement algorithms [4].

II. Related work

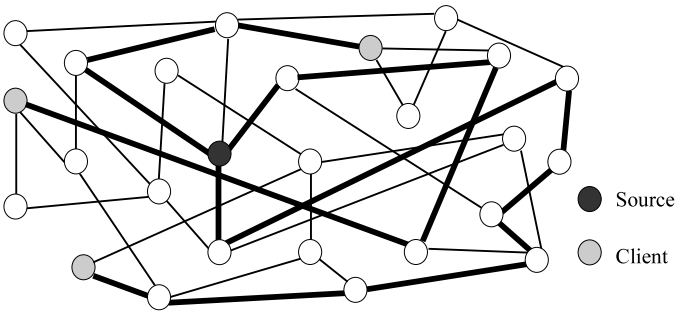
The RSP problem has been studied extensively in the literature. In [5] the objective was to minimize the cost of utilizing the servers and using the link band width, while serving requests based on their delay constraint. A database replication system that uses prior knowledge of query templates to select database table placements such that each query template can be treated locally is studied in [6]. In [7] authors have defined the QoS requirement in terms of the general distance metric, and investigated RSP in content distribution systems trying to meet the QoS requirements of clients while minimizing the replication cost. A two step algorithm that minimizes the client-to-replica latency in a wide-area network is proposed in [8]. In first step network regions where replicas should be placed are selected and in second step replicas are placed in different regions. In [9] authors have discussed and compared several procedures to place replicas in tree networks, subject to capacity, and QoS constraints. In [10] authors have proposed four natural heuristics and compared them numerical. The results show that the best results can be obtained with heuristics that have all the CDN servers cooperating in making the replication decisions. Ref. [11] have studied RSP problem in CDNs to meet the QoS requirements of clients while trying to minimize the replication cost which is defined in terms of storage, consistency management, or both of them. In [12] polynomial optimal solutions are applied to place a given number of servers in a tree network to minimize the average retrieval cost of all clients. Ref. [13] have studied the constrained mirror placement problem where the mirrors were allowed to be placed at some subset of network nodes only. It was shown that performance improvement after placing more mirrors beyond a certain number is not considerable. The detailed study of the problem of web server replica placement is presented in [14]. To make smart placement decisions, workload information such as client latency and rates of requests have been used to develop several placement procedures. Also authors have evaluated the performance of the these algorithms using both synthetic and real network topologies, as well as web server traces, and have shown that the placement of web replicas is crucial to CDN performance.

In this paper, compared to previous works, we consider a generalized RSP problem, which considers the latency restriction of the receivers, bandwidth, and storage constraints of the links and nodes. We must notice that the proposed placement algorithms are static in nature. However nothing precludes them from being classified as semi static placement algorithms [4].

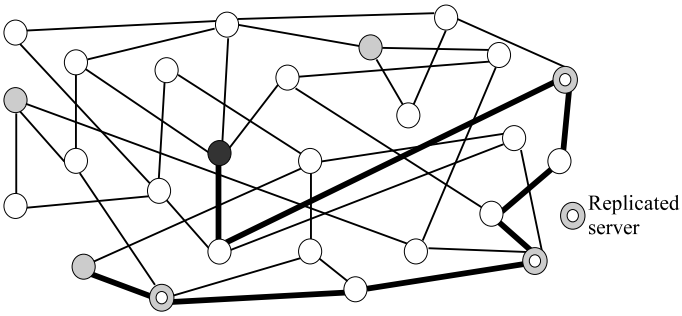
III. Problem formulation

Consider a large-scale network represented by a graph $G(V, E)$, where V is the set of nodes and E is the set of links. Let $C \subset V$, $S \subset V$, and $R \subset V$ denote the clients, sources, and replicated servers within the network, respectively. Assume that there are $|S|$ sources $S = \{S_1, ..., S_n\}$ and $|C|$ clients (receivers) $C = \{C_1, ..., C_l\}$ within the network, respectively.

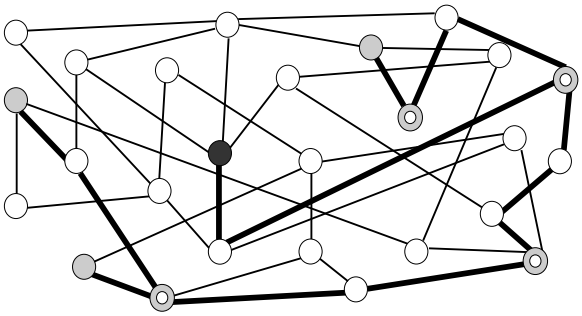
Each source, S_k, casts data at a specific rate, R_a, and each link (i, j) has a bandwidth $B(i, j)$. Moreover, the binary variable, y_{ijkl}, indicates if S_k sends data to clients using (i, j). To transfer a given amount of data from S_k to C_l using



(a) Searching for the shortest path between source and clients



(b) Choosing the path with highest latency, and placing replicated servers on it



(c) Finding the shortest path from clients to replicated servers placed in Step I, and placing replicated servers if needed

Fig. 1. Greedy algorithm

(i, j), the required bandwidth must be reserved on the link. Therefore, the following condition must always be satisfied:

$$\forall(i,j), \sum_{l \in C} \sum_{k \in S} (RB_k) \times y_{ijkl} \leq B(i,j), \tag{1}$$

where RB_k is the required bandwidth of source k.

We define the quality of service (QoS) requirements to be the time of retrieving data from G. Therefore, the designer must guarantee that at least one replicated server exists that has access cost less than the QoS (the latency constraint of C_l (L_i)) , and the replicated server has the copy of the data sent by S_k. Note that initially the network may not be able to fulfill the QoS constraints of C_l. One methodology to ensure fulfillment of QoS constraints is to replicate data within the network, that must ensure the following:

$$\sum_{t=1}^{n} X_{it} \times O_t \leq M^i, \forall(1 \leq i \leq n), \tag{2}$$

where M^i is the storage capacity of node i and $X_{it} = 1$ if node i holds a replica of object O_t and 0 otherwise. With data replication, better latency can be achieved that can be represented by:

$$\mathcal{L} = \sum_{k \in S} \sum_{l \in C} \sum_{i} \sum_{j} y_{ijkl} \times pd(i,j), \tag{3}$$

where $pd(i, j)$ is the propagation delay on (i, j). The RSP problem can formally be stated as follows: *Find the minimum number of replicated servers $|R|$ such that:*

$$min\{\mathcal{L} = \sum_{k \in S} \sum_{l \in C} \sum_{i} \sum_{j} y_{ijkl} \times pd(i,j)\}, \tag{4}$$

subject to the constraints of Eqs. (1), and (2).

IV. Placement algorithms

A. Greedy

One way to reduce the overall number of replicated servers is to identify a procedure that allows the reuse of resources. Such a procedure can be achieved by identifying the longest common path for each source and destination pair. Before we detail the greedy heuristic, we define the necessary variables. Let (RM_i), (CN_j) and $D(i,j) = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$ denote the storage required by S_i, clients that are not selected in **Step I of Algorithm 1**, and the distance between two nodes, respectively. Note that in line 6 of **Algorithm 1**, 2.85×10^8 refers to the wave propagation speed assuming a velocity factor of about 0.95 for the links [1]. The greedy algorithm consists of 3 steps that are explained in Fig. 1. The algorithm has a computational complexity of $\mathcal{O}(n.m^2.D(l+(l-1)^3.n)+n.l)$, where D is the computational complexity of the Dijkstra's algorithm ($\mathcal{O}(m^2)$) [15].

B. Greedy randomized adaptive search procedure (GRASP)

GRASP is a multi-iterative randomized heuristic. Each GRASP iteration contains two phases: (a) construction phase and (b) local search phase. To diversify the solution space we use a restricted candidate list (RCL) that contains paths that have latency greater than some percentage, α, of the maximum latency (Max_L), and follow the same procedure as in [3]:

$$RCL = \{L(path) \geq \alpha \times Max\_L\}, 0 < \alpha < 1. \tag{5}$$

Each GRASP iteration produces a solution. The best overall solution is kept as the result. Algorithm 2, constructs RCL using data from **Step I of Algorithm 1**, and performs local search. Contrary to **Step II of Algorithm 1**, GRASP considers the path in RCL chosen as explained above. The algorithm computational complexity of the aforementioned procedure is $\mathcal{O}(n.l.m^2.D + 3.n.l)$.

C. Path relinking

Figure 2(a) shows two example solutions for the RSP problem, which is the required initial step for a path relinking algorithm. In Fig. 2(b) a source-client pair (12 and 15, from the example solutions of Fig. 2(a)) is selected randomly and fixed if one of them is larger, by reusing the destination node. In this figure the destination node (node 15) of the source-client pair selected from Solution I, is reused. A new path between the selected source-client pair can be constructed using the method shown in Fig. 2(b). The 4th and 5th nodes of the new source-client pair may not be connected. Such an abnormality can be

TABLE I
RECEIVERS THAT CAN BE SERVED WITHOUT REPLICATED SERVERS

Number of sources and receivers	Receivers that ca be served without replicated servers for different QoS requirements		
	0.85 (time unit)	1 (time unit)	1.25 (time unit)
5 sources, 25 receivers	12%	20%	44%
10 sources, 40 receivers	42.5%	57.5%	72.5%
7 sources, 28 receivers	53.3%	53.3%	75%
4 sources, 20 receivers	15%	15%	15%
3 sources, 25 receivers	20%	20%	20%

fixed by connecting them by the shortest path that confirms to the latency constraints. In Fig. 2(b), assuming that there is no direct link between nodes 63 and 15 in the generated source-client pair, the problem is fixed by connecting them using the shortest path between them. By replacing the generated source-client pair in solutions of Fig. 1(a), two new solutions are generated. The aforementioned process is repeated several times until a maximum number of iterations has been reached, and finally we choose the solution with the minimum latency. More details on this algorithm can be found in [15].

D. Genetic Algorithm

Genetic algorithm has the following four primitive procedures: (a) generating initial population, (b) selection, (c) crossover, and (d) mutation. To apply genetic algorithm to our problem we start with a set of solutions, termed population and the solution is represented by a chromosome as shown in Fig. 2(a). Parent chromosomes are chosen randomly. After selecting chromosome pairs, a source-client pair is chosen randomly, and the crossover operation is performed as depicted in Fig. 2(b). Finally, to prevent solutions from converging into a local optimum, the mutation operation having the rate of 2% is applied. Figure 3 shows the mutation operation on a source-client pair. In this figure, the algorithm replaces the path from 38 to 43 with the shortest path between them. The aforementioned process is repeated several times until a maximum number of iterations has been reached, and finally we choose the solution with the minimum latency. More details on this algorithm can be found in [15].

Algorithm 1: Greedy

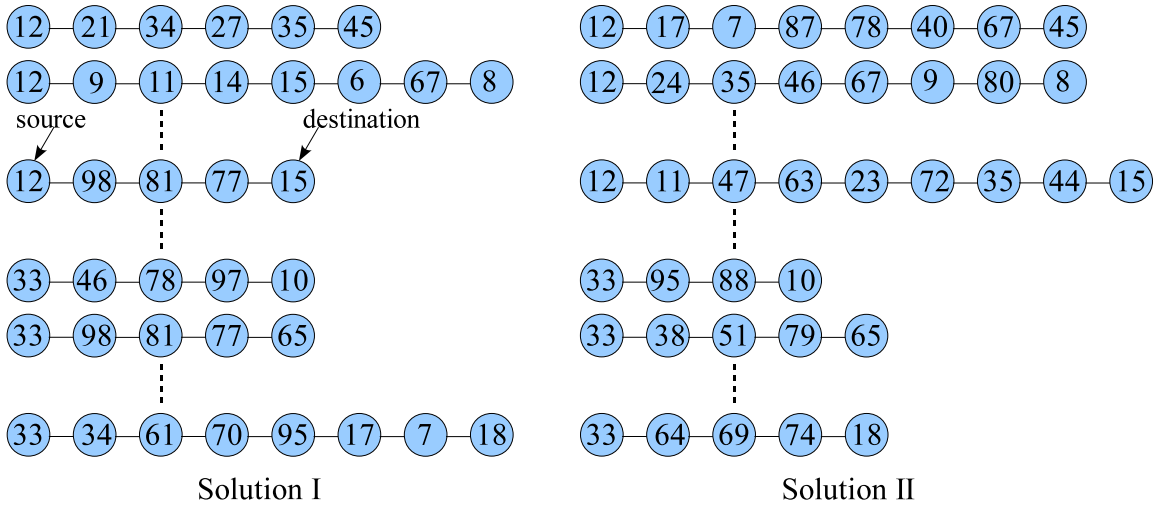
Step I:
for $\forall S_i \in S = \{S_1, ..., S_n\}$ **do**
  for $\forall C_j \in C = \{C_1, ..., C_l\}$ *of* S_i **do**
    for $q = 1$ **to** m **do**
      for $t = 1$ **to** m **do**
        $pd(q,t) = \frac{D(q,t)}{2.85\times10^8 m/s} + \frac{R_{ak}}{B(q,t)}$;
      end
    end
    Apply DIJKSTRA algorithm with this condition: $\forall(i,j), \sum_{m\in C}\sum_{k\in S}(RB_k) \times y_{ijkl} \leq B(i,j)$;
    $B(i,j) \leftarrow B(i,j) - RB_k$;
  end
end
Step II:
for $\forall S_i \in S$ **do**
  for $\forall C_j of S_i$ **do**
    $A \leftarrow$ The path with maximum cost;
    if Eq. (3) is not violated **then**
      $max\{\sum_i \sum_j y_{ijkl} \times pd(i,j)\} \leq QoS$;
      $A \leftarrow R_k$;
      $M(j) \leftarrow M(j) - RM_i$;
    end
  end
end
Step III:
for $\forall CN_j$ **do**
  $S \leftarrow CN$;
  $CN \leftarrow R$;
  call Step I of Algorithm 1;
  for $\forall S_i \in S$ **do**
    for $\forall C_j$ of S **do**
      $A \leftarrow$ The path with minimum cost;
      if Eq. (3) is not violated **then**
        $max\{\sum_i \sum_j y_{ijkl} \times pd(i,j)\} \leq QoS$;
        $A \leftarrow R_k$;
      end
    end
  end
end

V. EXPERIMENTAL RESULTS

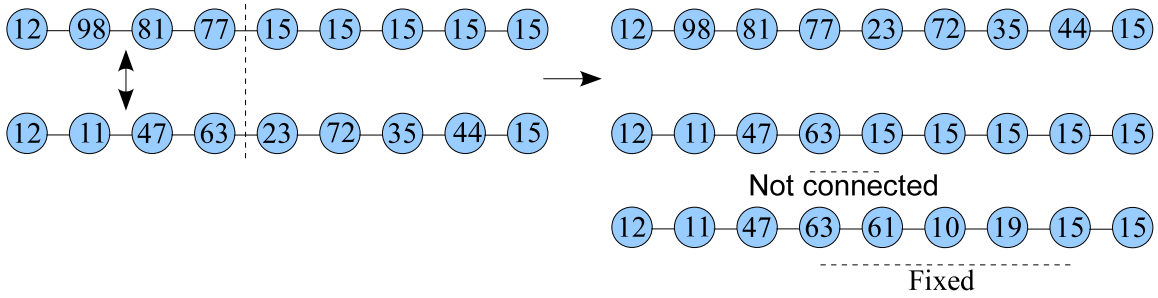
We used BRITE [17] to generate a network topology composed of 100 nodes as described in [5]. The proposed algorithms were applied to five sets of source-receiver combinations whose locations were chosen randomly. In GRASP, the α parameter was chosen to be 80% as suggested in [3]. We simulated the system for three different QoS requirements of receivers as previously described in Eq. (2). The QoS requirements were chosen proportional to propagation delay of communication links. The percentage of receivers within the system was selected randomly between 5% to 30% of the nodes. The aforementioned parameter was previously used in [18] and advocated in [19]. Table I, shows the percentage of the receivers that can be served without replicated servers. Table II summarizes results in terms of: (a) the number of replicas needed, (b) the average latency, (c) the number of

TABLE II
NUMBER OF REPLICAS, REUSED RESOURCES, AND THE AVERAGE LATENCY OF RSP ALGORITHMS FOR VARYING LATENCY CONSTRAINTS

Number of sources and receivers	Studied heuristics	Number of replicas needed for different QoS requirements			Average latency for different QoS requirements			Reused links for different QoS requirements			Reused storage for different QoS requirements		
		0.85 (time unit)	1 (time unit)	1.25 (time unit)	0.85 (time unit)	1 (time unit)	1.25 (time unit)	0.85 (time unit)	1 (time unit)	1.25 (time unit)	0.85 (time unit)	1 (time unit)	1.25 (time unit)
5 sources, 25 receivers	Greedy	40	36	35	1.5672	1.8539	1.5276	15	29	23	11	13	12
	GRASP	37	34	33	1.5583	1.8514	1.4866	16	30	25	12	14	16
	Path relinking	32	31	28	1.5088	1.8474	1.4415	18	18	27	15	16	19
	Genetic	30	28	27	1.4864	1.8203	1.4397	19	32	28	17	18	20
10 sources, 40 receivers	Greedy	47	45	43	0.9187	0.8460	0.7776	34	34	13	18	17	13
	GRASP	45	45	43	0.8961	0.8460	0.7776	35	34	28	19	17	13
	Path relinking	37	41	42	0.8562	0.8198	0.7573	42	35	19	42	19	15
	Genetic	×	×	×	×	×	×	×	×	×	×	×	×
7 sources, 28 receivers	Greedy	35	33	31	1.1603	1.0947	1.3364	33	18	21	14	13	12
	GRASP	33	32	27	1.0830	1.0667	1.0738	34	19	23	15	14	14
	Path relinking	30	31	26	1.0484	1.0631	1.0440	23	24	23	17	15	15
	Genetic	×	×	×	×	×	×	×	×	×	×	×	×
4 sources, 20 receivers	Greedy	37	33	31	2.0124	2.2461	2.1877	25	22	23	14	12	10
	GRASP	34	32	27	1.9808	2.0993	2.1083	26	24	24	15	13	11
	Path relinking	32	31	26	1.9348	1.9222	2.0583	27	25	25	16	16	12
	Genetic	×	×	×	×	×	×	×	×	×	×	×	×
3 sources, 15 receivers	Greedy	27	26	24	2.2856	1.9469	2.2233	27	21	22	13	9	9
	GRASP	25	25	23	1.9469	1.9642	2.0609	28	22	23	14	9	9
	Path relinking	23	21	19	1.9314	1.9160	1.8298	18	23	25	15	11	11
	Genetic	×	×	×	×	×	×	×	×	×	×	×	×



(a) Two example solutions



(b) Randomly selected source-client pairs and fixing the path if the 4th and the 5th nodes are not connected

Fig. 2. Path relinking algorithm

reused links, and (d) the number of the reused storage by the studied heuristics. The results of Table II are the average of the 135 simulations with a confidence interval of 95%. Note that in Table II, the number of receivers are a factor of 4, and 5 of the number of sources. This is due to the fact that the average Internet fanout is in the range of 3.8 to 4.9 [20]. Note that in Table Finally, II "×" denotes that the algorithm under consideration could not complete the test runs in reasonable amount of time. Table III shows the average run time of the studied heuristics.

VI. CONCLUSION

This paper formulated Constraint Replicated Server Placement in a Multicast Networks. For the aforementioned problem we proposed four heuristics that were benched marked using

BRITE. From the simulation results we concluded that the genetic algorithm identifies the best placement. However, the genetic algorithm suffers from slow execution time. Therefore considering solution quality and computational complexity we consider path relinking algorithm to be the ideal solution.

TABLE III
AVERAGE SIMULATION TIME

Latency requirements of receivers (time unit)	Average simulation time (seconds)
0.85	74.3027693
1	107.574794
1.25	96.5022219

Algorithm 2: GRASP

Call Step I of Algorithm 1;
for $i = 1$ **to** n **do**
  for $j = 1$ **to** l **do**
    $A(i,j) \leftarrow$ cost of paths;
  end
end
for $i = 1$ **to** n **do**
  for $j = 1$ **to** l **do**
    if $A(i,j) \geq \alpha \times max\_L$ **then**
      $RCL \leftarrow$ related path;
    end
  end
end
for $\forall S_i \in S$ **do**
  for $\forall C_j$ *of* S **do**
    $A \leftarrow RCL_i$;
    if Eq. (3) is not violated **then**
      $max\{\sum_i \sum_j y_{ijkl} \times pd(i,j)\} \leq QoS$;
      $A \leftarrow R_k$;
      $M(j) \leftarrow M(j) - RM_i$;
    end
  end
end
Call Step III of Algorithm 1;

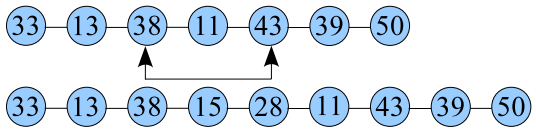


Fig. 3. Mutation

REFERENCES

[1] S. U. Khan, A. A. Maciejewski, and H. J. Siegel, Robust CDN Replica Placement Techniques, 23rd IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2009.

[2] A. Benoit,V. Rehn-Sonigo, and Y. Robert, Replica Placement and Access Policies in Tree Networks, IEEE Transactions on Parallel and Distributed Systems, 2008, Volume: 19, Issue: 12, pp 1614-1627.

[3] B. Li, F. Chen, and L. Yin, Server replication and its placement for reliable multicast, 9th International Conference on Computer Communications and Networks, 2000, pp 396-401.

[4] S. U. Khan, and I. Ahmad, Comparison and Analysis of Ten Static Heuristics-based Internet Data Replication Techniques, Journal of Parallel and Distributed Computing, 2008, Volume: 68, no: 2, pp. 113-136.

[5] G. Rodolakis, S. Siachalou, and L. Georgiadis, Replicated Server Placement with QoS Constraints, IEEE Transactions on Parallel and Distributed Systems, 2006, Volume: 17, Issue: 10, pp 1151-1162.

[6] T. Groothuyse, S. Sivasubramanian, and G. Pierre, Globetp: template-based database replication for scalable web applications, 16th international conference on World Wide Web, 2007.

[7] X. Tang, and J. Xu, QoS-Aware Replica Placement for Content Distribution, IEEE Transaction on Parallel and Distributed Systems, Volume. 16, no. 10, 2005.

[8] M. Szymaniak, G. Pierre, and M. Steen, Latency-Driven Replica Placemen, Symposium on Applications and the Internet (SAINT), 2005.

[9] A. Benoit,V. Rehn-Sonigo, and Y. Robert, Replica Placement and Access Policies in Tree Networks, IEEE Transactions on Parallel and Distributed Systems, 2008, Volume: 19, Issue: 12, pp 1614-1627.

[10] J. Kangasharju, J. Roberts, and K. Ross, Object Replication Strategies in Content Distribution Networks, Computer Communications, Volume: 25, no. 4, pp 367383, 2002.

[11] X. Tang, J. Xu, On replica placement for QoS-aware content distribution, 23th Annual Joint Conference of the IEEE Computer and Communications Societies, 2004.

[12] P. Krishnan, D. Raz, and Y. Shavitt , The cache location problem, IEEE/ACM Transactions on Networking, Volume. 8 Issue: 5, 2000.

[13] S. Jamin, A.R Cheng Jin Kurc, D. Raz, and Y. Shavitt, Constrained mirror placement on the Internet, 20th Annual Joint Conference of the IEEE Computer and Communications Societies, 2001.

[14] L. Qiu, V. Padmanabhan, and G. Voelker, On the Placement of Web Server Replicas, 20th Annual Joint Conference of the IEEE Computer and Communications Societies, 2001.

[15] M. Gendreau, and J. Potvin, Handbook of Metaheuristics, Springer science and Business media, 2010.

[16] D. Whitley, A Genetic Algorithm Tutorial, Statistics and Computing, pp: 6585, 1994.

[17] BRITE available at: http://www.cs.bu.edu/brite/.

[18] Z. Fei, M. Ammar, and E. Zegura, Efficient Server Replication and Client Re-Direction for Multicast Services, SPIE/ITCOM Conference on Scalability and Traffic Control in IP Networks, 2001.

[19] Z. Fei, M. Ammar, E. Zegura, Optimal allocation of clients to replicated multicast servers, 7th International Conference on Network Protocols, 1999.

[20] P. Radoslavov,R. Govindan, and D. Estrin, Topology-Informed Internet Replica Placement, 6th International Workshop on Web Caching and Content Distribution, 2001.