A Hybrid GRASP with Data Mining for Efficient Server Replication for Reliable Multicast

L.F. Santos, R. Milagres, C.V. Albuquerque, S. Martins and A. Plastino
Universidade Federal Fluminense – Instituto de Computação – Niterói – RJ – Brazil
{lfsantos,rpereira,celio}@ic.uff.br, {simone,plastino}@dcc.ic.uff.br

Abstract—Multicast communication is a topic of intense study by the network research community. The IP Multicast service of the network layer doesn’t provide the desired reliability to some multicast applications, and the interest towards approaches to reliable multicast communication has increased. In this work, we focus on the Server Replication method, wherein the data are replicated over a subset of the multicast-capable relaying hosts and retransmission requests from receivers are handled by the nearest Replicated Server. The problem of selecting the best subset of the multicast-capable relaying hosts to replicate the data is NP-Hard. We propose a hybrid metaheuristic to find near optimal solutions for this problem. This proposal is based on a hybrid version of the GRASP metaheuristic that incorporates data mining techniques. Experimental results show that our technique outperforms existing approaches.

I. INTRODUCTION

Multicast communication is the delivery of information to many receivers simultaneously. This information may come from just one source (one-to-many multicast) or many sources as well (many-to-many multicast). There are several potential applications of multicast, e.g., news delivery, stock quotes distribution, software updates, audio and video streaming, bulk data transfer, etc.

One could argue that current unicast network services are sufficient to support multicast. In fact, a straightforward implementation of a multicast transmission would be the creation of one unicast connection from the sender to each receiver. However, this strategy has a significant drawback: due to the transmission of multiple copies of the same data, this approach incurs in inefficient bandwidth usage. So, network services specially designed to multicast are needed to overcome this problem. The scheme used in current multicast services is the creation of a delivery tree, whose root represents the sender, leaves represent the receivers and internal nodes represent network routers or relaying servers. In this way, copies of the data just need to be created at split points of the tree.

Since the introduction of the IP Multicast model [1], interest in multicast increased, both scientifically and commercially [2]. A virtual network composed of multicast-capable routers was built to provide multicast services. This network was called MBone, the multicast backbone [3], and is continuously evolving. A number of protocols has been proposed, but there is no one considered the definitive solution yet [4].

Another issue regarding multicast is how to provide a reliable service. The service provided by the IP Multicast at the network layer follows a best-effort policy, so the delivery of all packets from the sender to receivers is not guaranteed. In the unicast case, the TCP transport protocol performs this task efficiently. However, in the multicast case, this task is substantially more complex. Many transport protocols were designed, each one targeting different requirements of multicast applications [5].

The provisioning of reliable multicast service deals with how to handle packet retransmissions. It is intuitive that a simple extension of the TCP, where the sender is responsible for the retransmission of lost packets, to multicast would overwhelm the sender, specially when the number of receivers is high. It has been shown that dividing the receivers in groups and performing a local recovery approach is a good strategy [6], [7]. In addition to reducing the workload at the sender, potential advantages of this method are the improvement of system throughput and reduction of bandwidth usage.

In this paper, we focus on the Server Replication approach, where the data are replicated at some of the multicast-capable relaying hosts and each of them is responsible for the retransmission of packets to receivers of its group. These relaying hosts are called Replicated Servers, or Repair Servers. The problem of selecting the best subset of the multicast-capable relaying hosts to act as Replicated Servers in a multicast scenario is a specialization of the \( p \)-medians problem, that has been proven to be NP-Hard [8]. Like any NP-Hard problem, heuristic-based approaches are valuable to enable the achievement of good quality solutions in reasonable time. Our contribution is a new heuristic-based algorithm for this problem. We propose a hybrid GRASP metaheuristic that incorporates data mining techniques to find good quality solutions for this NP-Hard problem. It is an improvement on the strategy adopted in [9].

The rest of the paper is organized as follows. In Section II, we give the mathematical formulation of the problem. In Section III, we give a brief review of the concepts behind our strategy and then present it. In Section IV, the experimental evaluations are presented. Concluding remarks are given in Section V.

II. PROBLEM FORMULATION

In this section, we review the formulation presented in [9]. We begin describing the function that will be used to model the cost of a multicast delivery tree. Then, the formulation is presented based on this cost function.
measures the cost of a multicast delivery tree. Hence, the objective is to find which multicast-capable relaying hosts must act as Replicated Servers to achieve a tree that minimizes this function.

Suppose that a node $s$ is the sender of a multicast session and the nodes in set $R$ are the receivers. In this case, a delivery tree $T$ is built with $s$ as the root, the nodes in $R$ as the leaves, and $MR$ internal nodes corresponding to the multicast-capable relaying hosts. There is a packet loss probability associated with each link of $T$. Now suppose that the data are replicated on $P \subseteq MR$ nodes. So $P$ is the subset of the multicast-capable relaying hosts that will be the Replicated Servers. Fig. 1 illustrates a delivery tree $T$ rooted at the sender $s$. The receivers, $R = \{r_1, r_2, ..., r_9\}$, are the leaves. The multicast-capable relaying hosts, $MR = \{n_1, n_2, n_3, p_1, p_2, p_3\}$ are the internal nodes. The set $P = \{p_1, p_2, p_3\}$ corresponds to the Replicated Servers.

Let $T_v$ denote the subtree of $T$ rooted at node $v$. The formulation we will present is based on the following terminology:

$R_v$: The set of receivers in $T_v$  
$R_v'$: The subset of elements in $R_v$ whose path between them and node $v$ does not include a Replicated Server  
$H_v$: The total number of links in $T_v$  
$P_v$: The set of Replicated Servers in $T_v$  
$P_v'$: The subset of elements in $P_v$ whose path between them and node $v$ does not include a Replicated Server

For example, consider $T_{n_1}$, the subtree rooted at node $n_1$ in Fig. 1. In this subtree, $R_{n_1} = \{r_1, r_2, r_3\}$, $R_{n_1}' = R_{n_1}$, $H_{n_1} = 4$, and $P_{n_1}' = P_{n_1} = \varnothing$. In $T_{n_2}$, $R_{n_2} = \{r_4, r_5, ..., r_9\}$, $R_{n_2}' = \{r_6\}$, $H_{n_2} = 9$, $P_{n_2} = \{p_1, p_2, p_3\}$, and $P_{n_2}' = \{p_1, p_2\}$.

The key idea of the formulation is to consider the cost of transmitting a packet as the product of its total number of transmissions (the original plus the subsequent retransmissions) and the number of links traversed until it reaches all receivers [6]. Hence, the tree in which this cost is minimal is the one that yields the best usage of the network resources, what incurs in the best performance of the multicast session.

Let $K_v$ denote the random variable representing the total number of transmissions of a packet from a node $v$ required to ensure its delivery to all receivers in $T_v$, given that the packet was successfully sent to $v$ from its parent. Note that in the case of the sender $s$, $K_s$ is the number of transmissions required to ensure the delivery of a packet to all receivers. If $T_v$ does not include Replicated Servers, every transmission of the packet will flow through every link of $T_v$. Hence, we can define the cost function of $T_v$ as follows:

$$cost(T_v) = E[K_v] \times H_v,$$

(1)

where $E[K_v]$ is the expectation of $K_v$. In the case that $T_v$ contains Replicated Servers, the sender $v$ will be responsible just for the delivery to nodes in $R_v' \cup P_v'$. It follows from the fact that every node $w \in P_v'$ is responsible for the transmission in $T_w$. Hence, the cost function can be defined as follows:

$$cost(T_v) = E[K_v] \times (H_v - \sum_{r \in P_v'} H_{r'}) + \sum_{r \in P_v'} cost(T_{r'})$$

(2)

Note that (1) is derived from (2) in the case of a tree without Replicated Servers. Note also that a subtree $T_w$ rooted at a node $w \in P_v'$ can also contain Replicated Servers. So, the above definition of the cost function is recursive.

To complete the formulation, it is still necessary to derive an expression to $E[K_v]$.

Let $p_n$ denote the packet loss probability over the link between $n$ and its parent. Consider a leaf node $n$ of a tree $T_v$. The probability that $k$ transmissions of one packet are sufficient in order to ensure its delivery to node $n$ from its parent, that is, the probability distribution function of $K_n$, is:

$$F_n(k) = P[K_n \leq k] = 1 - (p_n)^k$$

(3)

Note that the above equation can also be used for every node $w \in P_v'$. Since a node $w \in P_v'$ is responsible for the delivery in $T_w$, it is sufficient that packets arrive at $w$ to ensure their delivery to the receivers in $R_w$.

Now consider an internal node $n$ of a tree $T_v$. In this case, the probability distribution function of $K_n$ is:

$$F_n(k) = \sum_{l=0}^{k-1} \left( \binom{k}{l} (p_n)^l (1-p_n)^{k-l} \prod_{c \in \text{child}(n)} F_c(k-l) \right)$$

(4)

where $\text{child}(n)$ is the set of the $n$’s child nodes. The first term of the summation corresponds to the probability of occurring $l$ losses in $k$ transmissions to node $n$ from its parent. The second term corresponds to the probability that a packet will arrive at all $n$’s child nodes with at most $k - l$ transmissions from $n$ and then delivered successfully to all receivers in $T_n$. The summation considers all cases in which there is at least one successful transmission to $n$ from its parent.

For the root node $v$, the distribution function is the probability that a packet arrives at all of its child nodes with at most $k$ transmissions and then be delivered to all receivers:
\[ F_v(k) = \prod_{c \in \text{child}(v)} F_c(k) \quad (5) \]

In (6), we present the expression of \( E[K_v] \) in terms of its distribution function. Substituting this expression in (2) completes the formulation of the cost function \( \text{cost}(T_v) \).

\[ E[K_v] = \sum_{k=0}^{\infty} (1 - F_v(k)) \quad (6) \]

Note that the cost function can be easily generalized to the case where multiple senders co-exist. In this case, the multicast delivery trees, together, will form a graph \( G \). Given that routing and retransmission request delivery are independent for each sender, the cost function of the graph \( G \) can be defined as:

\[ \text{cost}(G) = \sum_s \text{cost}(T_s), \quad (7) \]

where \( \text{cost}(T_s) \) is the cost of the multicast delivery tree rooted at sender \( s \), given by (2).

Now the problem can be easily stated: for a given \( M \), find the best set \( P \subseteq MR \), where \(|P| = M\), so that the resulting multicast delivery graph has the minimum cost according to (7). This formulation is a specialization of the \( p \)-medians problem, that has been proven to be NP-Hard [8]. So, we can conclude that, as any NP-Hard problem, heuristic-based approaches are valuable to enable the achievement of good quality solutions in reasonable time.

### III. The Proposed Approach

Three heuristic algorithms were proposed in [9]. The first was a pure greedy strategy, the second was a genetic-based one and the other was a GRASP implementation. The experiments demonstrated that the GRASP achieved the best results in all cases, while the greedy algorithm was always the worst.

In this section, we present a new hybrid strategy based on the GRASP metaheuristic. Firstly, a brief overview of this metaheuristic is given. Next, the hybridization of this metaheuristic with data mining techniques is presented. Then, we describe the implementation of this approach to the problem stated earlier.

#### A. Greedy Randomized Adaptive Search Procedures

GRASP (Greedy Randomized Adaptive Search Procedures) [10] is a metaheuristic approach already applied successfully to several optimization problems [11].

It is a two-phase iterative process. The first phase of a GRASP iteration is the construction phase, in which a complete solution is built. Since this solution is not guaranteed to be locally optimal, a local search is performed in the second phase. This process is repeated until a termination criterion is met. The best solution found over all iterations is returned as the result. Fig. 2 contains the pseudo-code of the GRASP method for a maximization problem.

In the construction phase, a solution is built up from small parts. The components of the solutions are selected one by one and incorporated to the partial solution until it is completely built. In each step, the components not yet in the solution are ranked according to a greedy function. The better ranked components form a list, called Restricted Candidate List (RCL). The size of this list can be defined by a parameter \( \alpha \in [0, 1] \). Suppose that, in a maximization problem, the best ranked component has the cost \( \text{max}_s \text{cost} \). The RCL is then composed by all components which costs are in the interval \( [\alpha \times \text{max}_s \text{cost}, \text{max}_s \text{cost}] \). One component is randomly selected from this list and incorporated to the current solution. Note that this process would be purely greedy if the RCL was ever composed only by the best component (\( \alpha = 1 \)), and purely random if it was ever composed by all possible components (\( \alpha = 0 \)). The pseudo-code in Fig. 3 illustrates this process.

The solution obtained in the construction phase becomes that starting point of a local search loop. In this phase, the neighborhood\(^1\) of the solution is explored and, if a better solution is found, the local search is performed again, considering the neighborhood of this new solution. Otherwise, the local search terminates.

#### B. The hybrid GRASP

Our algorithm is based on a hybrid version of the GRASP metaheuristic that incorporates a data mining process. Recent applications of this method have achieved promising results [12], [13], [14].

Data mining refers to the extraction of new and potentially useful knowledge from datasets. The key motivation to incorporate a data mining process in GRASP is that it could be used to extract solution patterns that represent features of sub-

\[ \text{procedure} \text{ ConstructionPhase}() \]
\[ \text{1. } \text{sol} \leftarrow \varnothing; \]
\[ \text{2. repeat} \]
\[ \text{3. } \text{RCL} \leftarrow \text{BuildRCL(sol)}; \]
\[ \text{4. } \text{s} \leftarrow \text{SelectRandom(RCL)}; \]
\[ \text{5. } \text{sol} \leftarrow \text{sol} \cup \{s\}; \]
\[ \text{6. until SolutionCompleted(sol)}; \]
\[ \text{7. return sol}; \]

Fig. 3. Pseudo-code of the GRASP construction phase

\[^{1}\]A neighborhood of a solution \( s \) is defined by a function \( N(s) \) that relates \( s \) with a set of other solutions.
optimal solutions. Then, these patterns could be used to guide the search for better solutions.

The hybridization is defined as follows. In a first step, a number of GRASP iterations is executed to generate an elite set (Elite Set Generation Phase). The elite set contains the best solutions found over these iterations. The next step is to run a data mining algorithm to extract patterns in this elite set. The next GRASP iterations are executed with a slight difference in the construction phase: one of the patterns found is used as initial partial solution instead of starting the construction from an empty solution (Hybrid Phase). Fig. 4 illustrates the pseudo-code of the hybrid GRASP and Fig. 5, the adapted construction procedure.

C. The hybrid GRASP implementation

Our proposal is an extension of the GRASP implementation proposed in [9]. The solution of the problem is a set of $M$ multicast-capable relaying hosts that will be the Replicated Servers. The optimal solution is the one that minimizes (7). In the construction phase, relaying hosts are added to the solution one by one. In the case of just one sender, the relaying hosts are ranked according to (2). In the case of multiple senders, they are ranked according to the following equation:

$$\text{cost}(v) = \sum_s \text{cost}^s(v),$$

where $\text{cost}^s(v)$ is the cost of node $v$ (see (2)) in the context of the delivery tree of the sender $s$. Note that the selection of the relaying host with the highest cost as a Replicated Server would lead to the highest decrease in (7).

In the local search phase, the neighborhood structure considered is based on the $k$-exchange procedure. Every solution $n$ that can be obtained by replacing $k$ elements of a solution $s$ is part of $s$'s neighborhood. The value of $k$ was chosen to be 1, leading to a tractable algorithm complexity. In addition, another restriction is used. A relaying host $r$ can only be substituted by one of its one-hop neighbors, that is, relaying hosts that are connected to $r$ by a direct link. The best solution of the $s$'s neighborhood becomes the starting point of the next local search iteration.

We consider that a pattern of sub-optimal solutions is a set of relaying hosts that frequently appears in solutions of the elite set. In [12], the use of maximal frequent sets yielded the better solutions in average. A frequent set is maximal when no one of its supersets is also frequent. The algorithm FPmax* [15] is used to extract the maximal patterns from the elite set.

IV. Experimental Results

Multicast scenarios are quite diverse. They can exhibit different number of participants, number of senders, traffic volume, and so on. However, they can be roughly divided in two major groups: broadcasts and virtual conferences [16]. The former is characterized by having a single sender and a high amount of receivers. The latter has a smaller number of members, but they often act as senders and receivers as well.

The evaluations of the proposed strategy were made on five simulated multicast scenarios, where three of them represent virtual conferences and two represent broadcast transmissions. They were generated as follows. Firstly, the network topologies were created with the Georgia Tech Internetwork Topology Models (GT-ITM) toolkit [17]. Then a loss probability was assigned to each link of these topologies. After that, nodes were chosen to represent senders and receivers, and, finally, one multicast delivery tree was built to each sender.

Table I describes the simulated scenarios. For each of them, we describe its topology and the multicast session that run over this. The topologies are described by the number of transit nodes, the number of links, and the link loss probability range. The loss probability of all links are uniformly distributed in this range. The multicast sessions are described by the number of senders, receivers and candidate nodes, that correspond to the nodes that can be set as Replicated Servers. The candidate nodes are transit nodes that are part of at least one of the delivery trees.

![Fig. 4. Pseudo-code of the hybrid GRASP](image)

![Fig. 5. Pseudo-code of the adapted construction phase](image)
We have conducted experiments to compare the performance of our proposed strategy and the GRASP proposed in [9]. For the CONF₁, BROAD₁, and BROAD₂ scenarios, both strategies were run to find subsets of the candidate nodes, with sizes corresponding to 2.5%, 5%, 7.5%, and 10% of the number of transit nodes, to set as Replicated Servers. For the others, the subsets sizes corresponded to 5%, 10%, 15%, and 20%.

The algorithms were implemented in C++ and compiled with g++ 3.2.2. The tests were performed on a 1.7 GHz Intel Pentium 4 with 256 Mbytes of RAM.

Note that the cost function is based on a term that is an infinity summation (6). The implementations used an approximated value for this term. The computation of the summation was stopped when the last value of \( k \) evaluated didn’t contribute to the summation with more than \( 10^{-4} \).

Both implementations were run ten times for each problem instance, with different random seeds. The GRASP parameter \( \alpha \) was set to 0.7. Each run consisted of 500 iterations. In the hybrid GRASP, the Elite Set Generation Phase took 250 iterations and the Hybrid Phase took the rest. The size of the elite set, from which the patterns are mined, was set to 10. A set of nodes was considered a pattern if it was present in at least 2 of the elite solutions. The ten largest patterns found were used.

The solution costs obtained are shown in Table II. The first two columns describe the problem instances, showing the multicast scenario and the number of nodes to be set as Replicated Servers. The next three columns contain the results achieved by the GRASP proposed in [9] (GRASP), and the final three columns, the results of our strategy (DMGRASP). For both strategies, we show the best, average, and standard deviation of the solution costs obtained over the ten runs. Analyzing the best results, we can note that the GRASP was never superior than DMGRASP. Out of 20 instances, the DMGRASP achieved better solutions in 12, and the solutions were the same in 8. Comparing the average results, the DMGRASP was superior in 13, the GRASP in 4, and they tied in 3.

Table III shows the computational time analysis. For each strategy, we show the average and standard deviation of the execution time taken by the ten runs. DMGRASP was considerably faster in all tests. The last column shows the average time reduction of DMGRASP in relation to the GRASP. Considering the average of all instances, DMGRASP was 36.8% faster than GRASP. The main reason for this improvement is that the use of patterns leads to the construction of better initial solutions to the local search phase. This incurs in less time taken to converge to a local optimal solution.

We can also observe from the results of Table II that the multicast sessions cost does not decrease linearly with the increase of the number of Replicated Servers. To further analyse the behavior of the multicast sessions cost, we have run the DMGRASP for the above multicast scenarios varying the number of Replicated Servers. In the graphic of Fig. 6, we plotted the costs corresponding to some percentages of the
candidate nodes set as Replicated Servers. Note that the cost converges to an asymptotic value. It means that including a new Replicated Server is recommended only when the benefits compensate the expenses involved.

V. CONCLUSIONS AND FUTURE WORK

The Server Replication approach is a successful technique to enable efficient reliable multicast transmission. In this work, we proposed a hybrid GRASP metaheuristic that incorporates a data mining process for the problem of finding the best multicast-capable relaying nodes of a multicast scenario to set as Replicated Servers.

Computational results demonstrated that the proposed strategy has found, in general, better solutions than the GRASP proposed in [9], and, specially, was 36.8% faster, what is a considerable save in execution time.

We believe that the results achieved could be even better. In the experiments, the Elite Set Generation Phase has taken 250 iterations. We noted that, in general, the pool quality stabilized well before these iterations finish. The use of a criterion to automatically find the ideal number of iterations for this phase would avoid the execution of useless iterations. It means that the execution times could be even lower.

Another improvement would be the alternation of GRASP iterations and the data mining procedure, instead of mining the patterns just once. This may allow the extraction of patterns from refined elite solutions.

A trend in optimization research is the parallelization of algorithms. The use of many processors can not only reduce execution times, but improve solutions obtained. We are currently working on parallelization strategies for the hybrid GRASP metaheuristic.

REFERENCES