Fast Heuristics for the QoS Multicast Routing Problem

Alia Bellabas, Samer Lahoud, Miklós Molnár

Fast Heuristics for the QoS Multicast Routing Problem

Alia Bellabas
IRISA, INSA Rennes
Campus de Beaulieu, Rennes 35042
Email: alia.bellabas@irisa.fr

Samer Lahoud
IRISA, University of Rennes 1
Campus de Beaulieu, Rennes 35042
Email: samer.lahoud@irisa.fr

Miklós Molnár
LIRMM, University of Montpellier 2
161 Rue Ada, Montpellier 34095
Email: miklos.molnar@lirmm.fr

***Abstract*—The NP-hard QoS Multicast Routing (QoSMR) is one of the most challenging problems in recent networks. In this paper, we investigate the QoSMR problem as well as its different existing formulations. We analyze the limitations of these formulations and propose a novel one. The main advantage of our formulation is that it considers not only the quality of the multicast subgraph formed by the computed paths, but also takes into account the end-to-end quality of each of these paths. Moreover, we study the state of the art proposal for solving the QoSMR problem and show that the well-known MAMCRA algorithm can be very expensive in computation time. Therefore, we propose two efficient heuristics based on the computation of shortest paths. Extensive simulations are performed, and obtained results prove that these fast heuristics have bounded computation time and find satisfying solutions for the QoS requirements.**

I. Introduction

Quality of Service (QoS) routing known as multi-constrained routing consists in computing paths that meet a set of requirements such as delay, bandwidth, and cost. Most of the emerging multimedia applications such as video broadcasting and online gaming become more stringent with quality of service and require multicast routing. Multicast routing aims to transmit packets simultaneously from one source node to multiple receivers.

It has been proved that the QoS Multicast Routing (QoSMR) problem is NP-hard [10]. There exist different approaches in the literature for solving this problem. A first approach aims to solve a Single-objective Optimization Problem (SOP) constructed from the initial QoSMR. A second approach solves the QoSMR problem as a multicast Multi-Constrained Optimization Problem (MCOP).

The main idea of the first approach is to solve a SOP constructed from the initial QoSMR, and then recover the solutions for the initial problem. In order to obtain a SOP from the QoSMR problem, a first technique transforms one constraint into a metric to optimize, like the optimization of the cost under the delay constraint [4]. Another technique proposes a linear combination of the metrics as done in [5]. Using one of the above cited techniques, the QoSMR problem is replaced by the constructed SOP. The Shortest Path Heuristic (SPH) [13] is one of the proposed algorithms to solve the resulting problem. SPH starts by computing the shortest path between the source node and the closest one of the multicast group. Then, it joins the next closest node to the current path and forms a tree. The algorithm reiterates until all the destinations have joined the tree.

A second approach solves the QoSMR as a multicast MCOP using a special length function. Many works propose to extend the most pertinent research results on the multi-constrained unicast routing to the multicast MCOP. H_MCOP is one of the well-known multi-constrained unicast algorithms that was introduced in [2]. This algorithm is based on the execution of two modified versions of Dijkstra's algorithm in forward and backward directions to compute the shortest paths between two nodes. To solve the multicast MCOP using H_MCOP, many algorithms were proposed. In [1], the authors define an algorithm based on SPH and H_MCOP. In [3], the authors propose an algorithm that searches for a feasible solution to the problem by first finding a feasible partial tree that spans the source and some of the destinations. Then, it builds up the remaining destinations using H_MCOP.

The state of the art presents the Multicast Adaptive Multiple Constraints Routing Algorithm (MAMCRA) as one of the most pertinent algorithms for the MCOP. In fact, the above cited algorithms aim to compute *a tree as a solution*, and this leads to the constrained Steiner Minimal Tree. However, the optimal solution is not *always* a tree. MAMCRA is an algorithm that solves the MCOP by computing *a multicast subgraph*. For this, MAMCRA starts by computing optimal paths between the source node and the set of the destination nodes using a modified version of SAMCRA [8], an exact multi-constrained unicast routing algorithm. The computed paths may form loops; therefore, MAMCRA uses a second step to eliminate some of these loops using a greedy algorithm. MAMCRA is pertinent since it always finds a solution for the QoSMR problem if such a solution exists. However, MAMCRA can be expensive in computation time [9].

In this paper, we investigate the different formulations of the QoSMR. An interesting non linear length function has been introduced in [8] to solve the multi-constrained unicast routing. This length expresses the value of the most critical metric of a path regarding the end-to-end requirements. The same length function was proposed to formulate the QoSMR problem in [7]. As we show in the following, the use of such a length function is less pertinent to evaluate the quality of a multicast subgraph. Indeed, most of the known QoSMR formulations

consider the subgraph proposed for the multicast routing as a unique entity and try to reduce its total length without considering the end-to-end requirements of each destination. Therefore, we propose a novel formulation of the problem by defining a new length function for multicast routing structures that takes into account the quality of each of the end-to-end computed paths. This formulation is interesting to evaluate the quality of the computed multicast subgraphs.

In addition, efficient algorithms with bounded execution time are required to solve the general QoSMR problem. In this paper, we propose two fast heuristics based on the constructed SOP approach. These heuristics compute shortest paths by adapting the Yen's algorithm [6]. Note that the Yen's algorithm has the smallest combinatorial complexity for computing the shortest paths [11]. The paths are computed using one additive metric, from the source to the destinations in an increasing order of length. The computation stops when a feasible path is found or a given upper bound of the computed paths is reached. This upper bound is used to limit the complexity of the algorithms. The difference between the two heuristics lies in the metric used in the shortest path computation. The Hop Count Approach (HCA) considers the hop count metric, while the second heuristic named Metric Linearization Approach (MLA) combines the QoS metrics into one weighted one. After shortest paths computation, the resulting set of paths may contain useless loops. Thus, we propose to apply the same greedy algorithm as MAMCRA, to remove the loops. To evaluate the performance of our heuristics, we perform extensive simulations. The obtained results prove that the usage of our heuristics provides guaranteed execution time while giving satisfying solutions, considering our proposed formulation.

II. Problem Formulations

The network is modeled as an undirected weighted graph $G(N, E)$, where N is the set of nodes and E the set of links. Each link e is characterized by a weight vector $\vec{w}(e) = (w_1(e), w_2(e), ..., w_m(e))$, where $w_i(e)$ corresponds to the weight associated to the QoS metric i. The QoS metrics can be classified into additive metrics such as delay, multiplicative metrics such as loss rate or bottleneck metrics such as available bandwidth. In the following and without loss of generality, we only consider additive metrics. The QoSMR problem consists in computing paths from a source node s to a set of r destination nodes $D = \{d_1, d_2, ..., d_r\}$, while satisfying m constraints given in a constraint vector $\vec{L} = (L_1, L_2, ..., L_m)$. We denote by $M(\{s, D\}, H)$, with $H \subseteq E$, the multicast subgraph formed by the computed paths. The length of a path $p(s, d_j)$ corresponding to the metric i is given by $l_i(p(s, d_j)) = \sum_{e \in p(s,d_j)} w_i(e)$. Thus, we define a feasible path $p(s, d_j)$ as follows:

$$l_i(p(s, d_j)) \leq L_i, \ \forall i = 1, ..., m \tag{1}$$

Using the Pareto dominance, a path $p(s, d_j)$ dominates a path $p'(s, d_j)$ if:

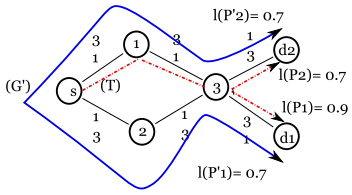

Fig. 1: Relevance of the length function choice

$$\begin{cases} l_i(p(s, d_j)) \leq l_i(p'(s, d_j)), \ \forall i = 1, ..., m \\ l_j(p(s, d_j)) < l_j(p'(s, d_j)), \text{for at least one metric } j \end{cases} \tag{2}$$

To evaluate the quality of a path $p(s, d_j)$, an interesting non linear length function was defined in [8]:

$$l(p(s, d_j)) = max_{i=1,...,m}\left(\frac{\sum w_i(e)_{e \in p(s,d_j)}}{L_i}\right) \tag{3}$$

This length function considers the value of the most critical metric of a path regarding the end-to-end requirements. Based on an extended version of this length function, Kuipers *et al.* give three formulations of the QoSMR in [7]:

Formulation I- Multiple Constrained Multicast: The problem consists in finding a multicast subgraph $M(\{s, D\}, H)$ such that each path $p(s, d_j)$ in M is feasible.

Formulation II- Multiple Parameter Steiner Tree: This problem aims to find a multicast subgraph $M(\{s, D\}, H)$ with a minimal length $l(M)$, where l is defined by:

$$l(M) = max_{(i=1,...,m)} \frac{\sum_{e \in M} w_i(e)}{L_i} \tag{4}$$

Kuipers *et al.* prove that the solution of Formulation II is always a tree. They have also shown that this tree does not necessarily satisfy the QoS constraints for all destinations. Therefore, they define a third formulation, which is a combination of Formulation I and Formulation II.

Formulation III- Multiple Constrained Minimum Weight Multicast (MCMWM): This problem consists in finding a multicast subgraph $M(\{s, D\}, H)$ such that each path $p(s, d_j)$ in M is feasible and $l(M)$ is minimum.

The last formulation proposes the minimization of the total length of the partial spanning subgraph without violation of the QoS constraints. However, we detect a limitation in the used length function, which can influence considerably the quality of the computed multicast subgraph. Indeed, we notice that this length function mainly focuses on the evaluation of the computed multicast subgraph without considering the quality of the end-to-end paths. To illustrate this limitation we use the example presented in Figure 1.

In this example, we apply Formulation III. The constraint vector $\vec{L}$ is given by: $(10, 10)$. For the multicast group $\{s, \{d_1, d_2\}\}$, we notice that there are three set of paths to reach the destinations $\{d_1, d_2\}$ from s. A tree T, traversing node 1, with length $l(T) = 1$, illustrated with dashed lines, a tree T', traversing node 2, with length $l(T') = 1$, which we do not illustrate here for simplicity, and a multicast subgraph G' with length $l(G') = 1.2$, illustrated with solid bold line. With Formulation III, the solution of the problem will be the

tree T. However, if we consider the end-to-end quality of the computed paths, we notice that the length of P_1' is shorter than that of P_1, while P_2 has the same length as P_2'. Hence we can notice that comparing G' to T, the total length of the subgraph increases, while the length of the unicast paths decreases.

To conclude, we have shown in the previous example that the total length of the subgraph and the length of each of the paths may evolve in different ways. In order to simultaneously characterize the quality of the multicast subgraph and the quality of each of the unicast paths, we propose in the following a more appropriate formulation based on a new length function defined by:

$$l'(M) = max_{(i=1,...,r)}(l(p(s,d_i)) \quad (5)$$

The length function l' evaluates the quality of the computed multicast subgraph M based on the end-to-end properties of its paths. In fact, the length of M corresponds to the length of the most critical unicast path. So, if the furthest destination to be reached has its QoS requirements satisfied, it is clear that the other destinations are also satisfied. This length can be considered as the *non linear diameter* of the computed multicast subgraph M.

Formulation IV- Adapted Multiple Constraints Multicast (AMCM): It aims to find a multicast subgraph $M(\{s,D\},H)$ formed by feasible paths $p(s,d_j)$, $d_j \in D, j = 1,...,r$, with a minimum length l' given in Equation (5).

III. Multicast Route Computation with MAMCRA

A. Description of the MAMCRA Algorithm

Referring to the state of the art, MAMCRA is an efficient algorithm for solving the QoSMR problem. This algorithm is pertinent since it *solves* Formulation I and *approximates* Formulation III. Given a multicast group $\{s,D\}$, MAMCRA computes a multicast subgraph $M(\{s,D\},H)$ that gives for each node pair (s,d_j) a path with minimal non linear length in Equation (3) that satisfies a set of QoS constraints, if such a path exists. MAMCRA operates in two steps as follows.

Step A: this step computes the paths from the source s to the destinations $d_i \in D$ using a modified version of SAMCRA [8], an optimal multi-constrained unicast routing algorithm. It begins by exploring the neighbors of the source s and chooses the closest node using the non linear length function in Equation (3). At each iteration, the algorithm chooses the closest neighbor of the current explored node. The dominated paths are regularly dropped, while all non dominated ones are memorized. Step A ends when all the destinations $d_i \in D$ are reached, and there is no possibility to find a better path for any destination. We notice that the computed paths may contain loops.

Step B: the second step of MAMCRA uses a greedy algorithm to eliminate the loops generated in *step A*. Assuming that, some intermediate nodes are reached from the source by multiple paths. The greedy algorithm tries to keep one path toward the shared node and deletes all the others if there is no violation of the end-to-end constraints. This process is undertaken progressively until all paths are treated once. Because of the greedy aspect of the used algorithm, the resulted multicast subgraph may not be necessarily the optimal one considering Formulation III neither considering Formulation IV.

B. Shortcoming of MAMCRA Algorithm and Motivation for the Fast Heuristics Proposal

It has been proved that the multi-constrained routing problem is NP-hard either in the unicast case [10], or in the QoSMR one [7]. Note that MAMCRA is an efficient algorithm that solves Formulation I and approximates Formulation III. As explained in section II, the last formulation suffers from the biased evaluation of the quality of the computed multicast subgraph. Therefore, we reformulate the QoSMR in order to aptly evaluate the efficiency of the algorithms proposed in this paper as well as MAMCRA. Despite its effectiveness in solving Formulation I, MAMCRA can be expensive with a combinatorial complexity in $O(k|N|\log(k|N|) + k^2m|E| + |N|r^2)$ [9], with k the number of paths that are stored at each intermediate node during the computation. Moreover, as MAMCRA provides an approximation for Formulation III, especially after reducing the loops, it may be more interesting to use a heuristic with less combinatorial complexity. This summarizes our main motivation for introducing, in the followings, two fast heuristics.

IV. Proposed Heuristics

To solve the QoSMR problem, we propose two heuristics based on the computation of k-shortest paths. We argue that one of these k-shortest paths would be feasible, with a reasonable low value for k. Like MAMCRA, the proposed heuristics are used for static schemes. However, these heuristics have a combinatorial complexity that is bounded by the maximal number of computed shortest paths k_{max}, and this allows the fast re-computation of paths if necessary. The proposed heuristics use one additive metric. The first heuristic computes the paths with the smallest number of hops. The second heuristic uses a combination of the QoS metrics in a single one, as it is explained in section IV-B. The proposed heuristics are based on Yen's algorithm, which we outline in the following.

A. Yen's Algorithm

As shown in many studies [11], Yen's algorithm is the most pertinent *k* shortest paths algorithms. This algorithm was introduced in [6].

For a given node pair (s,d_i) and a given integer k, this iterative algorithm computes, using a single additive metric, the k shortest paths between these nodes, if such paths exist. For that, it begins by computing the first shortest path. At the i^{th} iteration, the algorithm computes the i^{th} shortest path by considering all possible paths that *deviate* from the $(i-1)^{th}$ shortest path, without considering paths that are not already computed. For instance, let us consider the example in Figure 2, where the first two shortest paths between the nodes 1 and 4 are to be computed. The algorithm begins by computing the shortest path P_1. Then, it computes all shortest paths that deviate from P_1 at nodes 1 and 2; Thus, two paths P' and P''

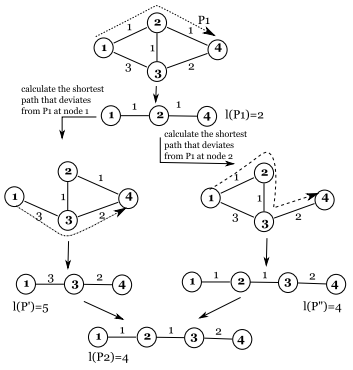

Fig. 2: An example of Yen's algorithm processing

are computed. Then, the shortest one, here P'' with length 4, is the second shortest path P_2.

B. Algorithmic Description of the Proposed Approaches

To solve the QoSMR problem, we propose fast heuristics that compute shortest paths in an increasing order, for each pair of nodes (s, d_j), where d_j is a multicast destination. These heuristics stop when a feasible path is found for each node pair, or an upper bound of the number of computed paths k_{max} is reached. The proposed heuristics use a single additive metric obtained by two different ways:

Hop Count Approach (HCA): in this approach, the algorithm computes shortest paths considering the number of hops as a single metric. The combinatorial complexity of HCA is in $O(r * k_{max}|N|(|E| + |N|log|N|))$, where r is the number of multicast destinations.

Metric Linearization Approach (MLA): this approach first substitutes the weight vector $\vec{w}\ (e) = (w_1(e), .., w_m(e))$ by a scalar weight $w^{'}(e) = \sum_{i=1,...,m} \alpha_i w_i(e)$. To calculate the parameters α_i, MLA computes $p_i^*(s, d_j),\ i = 1, ..., m$, the paths obtained by considering respectively a single additive weight $w_i(e)$ on each link e. Then, α_i is obtained by:

$$\alpha_i = \frac{l_i(p_i^*(s, d_j))}{L_i} \tag{6}$$

α_i is called *the criticality degree* of the constraint L_i. When α_i is close to 1, the length of the path $p_i^*(s, d_j)$ is close to the constraint L_i. Consequently, it is important to satisfy at first the constraint L_i that is the most critical. After the initial weights $\vec{w}\ (e)$ are replaced by the new scalar ones $w'(e)$, the MLA approach computes shortest paths considering these new weights.

In addition to the combinatorial complexity of HCA, MLA needs $O(r * m(|E| + |N|log|N|))$ operations to compute m shortest paths considering the m metrics separately from the source s to each destination d_j, and $O(r * |E|)$ operations to combine the weights of the graph links. The complexity of MLA is then in

$$O((r * m + k|N|)(|E| + |N|log|N|) + r * |E|) \tag{7}$$

Our heuristics HCA and MLA process as follows. For a given graph G, a given multicast group $\{s, D\}$, a constraint vector $\vec{L}$, the algorithm returns a multicast routing subgraph by computing for a given node pair (s, d_j) the first shortest path that satisfies $\vec{L}$. If a destination is not reached after k_{max} iterations, it is removed from the destination set D. The process stops after all paths are computed, and all non reachable destinations removed. A meta-code is presented in Algorithm 1 for the MLA approach.

Algorithm 1 MLA meta-code

```
for (j = 1, ..., r) do
  Compute p_1^*(s,d_j), p_2^*(s,d_j), .., p_m^*(s,d_j) the shortest paths considering respectively the m metrics
  for (i = 1, ..., m) do
    α_i = (Σ_{e∈p_i^*(s,d_j)} w_i(e)) / L_i  the criticality degree of L_i
  end for
  w'(e) = Σ_{i=1,...,m} α_i w_i(e),  ∀e ∈ G
  FeasiblePath← false, k = 1
  while ((FeasiblePath≠ true) and (k ≤ k_max)) do
    Compute p_k(s,d_j) the k^th shortest path between s and d_j
    if p_j(s,d_i) is feasible then
      FeasiblePath← true
    else
      k ← k + 1
    end if
  end while
end for
```

V. Performance Evaluation and Simulations

The performance of the proposed heuristics and the MAMCRA algorithm are investigated through extensive simulations. For this, we use a realistic network with 50 nodes and 82 links denoted by Real-Topology [12]. Each link is associated with two additive weights generated randomly using a uniform distribution in the interval $[1, 1024]$.

Different classes of constraints are also considered, from strict constraints to loose ones. The constraint vectors are generated in a way that they browse a defined generation space by areas from the strictest constraints to the loosest ones. In Figure 3, P_1 and P_2 denote the shortest paths that minimize the first and second metric respectively. The colored rectangle (B) delimited by $(l_1(P_1),\ l_2(P_1))$ and $(l_1(P_2),\ l_2(P_2))$ circumscribes the region where the constraints are selected. Ten areas: area 1 to area 10 are considered within (B). The constraints are selected randomly within these areas. Outside the specified region, the QoS constraints are less interesting to be examined. Indeed, all constraints that are generated within space (A) are infeasible, while constraints generated in space (C) are trivial and a polynomial algorithm will be sufficient to compute the multicast subgraph. We note that strict constraints are close to $l_1(P_1)$ and $l_2(P_2)$ (area 1), while loose constraints are close to $l_1(P_2)$ and $l_2(P_1)$ (area 2).

A first series of simulations has been performed. We randomly generated 100 instances of link weights. For each instance of link weights, 100 multicast groups are randomly

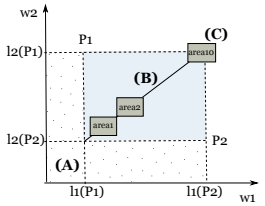

Fig. 3: Constraint generation areas

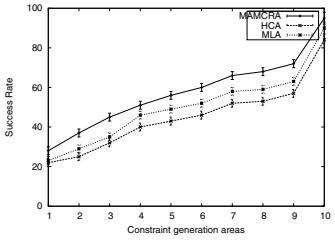

Fig. 4: Success Rate in Real-Topology

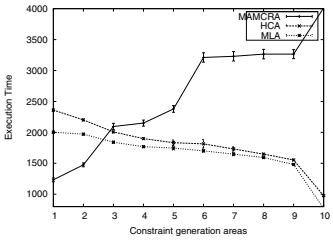

Fig. 5: Execution Time in Real-Topology

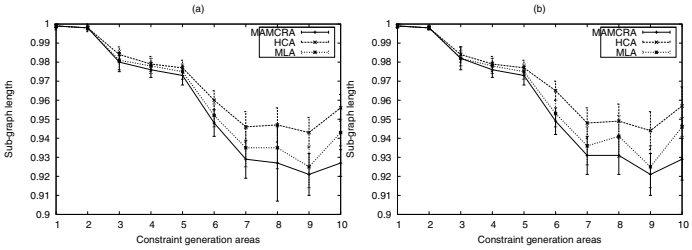

Fig. 6: Multicast subgraphs lengths (a) before and (b) after eliminating loops in Real-Topology

selected. The multicast group size is fixed to 50% of the total number of nodes. Thereafter, ten routing requests are generated in each area, from the strictest constraints (area 1) to the loosest ones (area 10). After that, the three algorithms: MAMCRA, HCA and MLA are executed independently to find a solution. Four performance measures are computed.

- *Success rate:* corresponding to the number of satisfied routing requests from 100 generated requests,
- *Quality of computed multicast subgraphs:* corresponding to the new length defined in Equation (5), which represents the diameter also called the length of the computed multicast subgraphs. This length is calculated before, then after eliminating the loops,
- *Execution time:* is the number of operations[1] that are needed to respond to a given multicast group request, either by a successful computation or by a failure,
- *Number of loops:* the number of detected loops including the number of deleted ones after the second step of the algorithms.

We note that these performance measures have been computed with 95% confidence intervals according to the ten constraint generation areas. We also note that the upper bound of the computed shortest paths in our heuristics, i.e. k_{max}, is fixed to three. However, our current works study the influence of k_{max} value in the proposed heuristics.

A. Success Rate

Figure 4 shows the success rate of the simulated algorithms: MAMCRA, HCA and MLA. Foremost, we notice that the success rate of the three algorithms is increasing. In fact, for strict constraints there are few feasible requests, and this number is increasing when constraints become loose. Since MAMCRA is an exact algorithm, it gives the highest success rate. The success rate of MAMCRA varies from 27% for strict constraints to 96% for loose ones. The gap between MAMCRA and our heuristics is 7% for MLA and 10% for HCA for strict constraints. For loose constraints, the difference does not exceed 5% with MLA and 8% with HCA. We notice that the success rate of our heuristics can be closer to MAMCRA if k_{max} is increased. In fact, when $k_{max} \rightarrow +\infty$ HCA and MLA become exact algorithms and thus always find feasible solutions.

[1]An elementary operation corresponds to the visit of one node

B. Execution Time

In Figure 5, we notice that the execution time of MAMCRA is increasing following the constraint areas, while HCA and MLA are decreasing. For strict constraints, the execution time of MAMCRA is less than HCA and MLA. Indeed, MAMCRA removes rapidly the non feasible paths, while HCA and MLA return the non feasibility response for a given request only after they compute the three shortest paths. When the constraints become less strict, HCA and MLA find solutions before computing the three shortest paths. For this, the execution time of HCA and MLA becomes until five times smaller than MAMCRA.

C. Quality of Computed Multicast Subgraphs

Figure 6 shows the length of the computed multicast subgraphs (a) before and (b) after eliminating loops. In Figure 6.(a), MLA and HCA give solutions with lengths very close to MAMCRA, especially for strict constraints. Indeed, when constraints are strict, few paths are feasible. Thus, when HCA and MLA find feasible paths, they find the optimal ones. For loose constraints, a request may have many feasible paths for each destination, and the proposed heuristics stop when they find the first feasible one to each destination. Thus, these paths may not be necessarily the optimal ones. However, the lengths of the computed paths are still close to those found by MAMCRA and the difference between the lengths of the computed multicast subgraphs does not exceed 3.12% for HCA and 1.72% for MLA. We notice that after eliminating

Number of	area1	area10
Detected loops	MAMCRA: 13.42 ± 3.21 HCA: 8.01 ± 1.63 MLA: 10.39 ± 1.65	MAMCRA: 22.68 ± 3.53 HCA: 14.84 ± 2.47 MLA: 20 ± 2.01
Deleted loops	MAMCRA: 1.15 ± 0.16 HCA: 0.8 ± 0.14 MLA: 1.09 ± 0.15	MAMCRA: 1.88 ± 0.16 HCA: 1.54 ± 0.17 MLA: 1.8 ± 0.17

TABLE I: Number of detected and deleted loops in Lattice-Topology

loops, the difference between the lengths is very small for each algorithm. However, the elimination of some loops can significantly increases the length of the computed multicast subgraph. In fact, MLA computes paths according to the criticality degree of the constraints. Therefore, if a prefix[2] that gives good solution for the first metric is replaced by another prefix that optimizes the second metric, the resulting path will be longer in terms of the non linear length. We can also notice that the length of the computed multicast subgraphs is increasing at the area 10 for the three algorithms. This is due to the increasing number of requests that become feasible within this area ($3/4$ of the area is situated in the trivial region (C)).

D. Number of Detected and Deleted Loops

As shown above, the diameter of computed multicast subgraphs slightly changes after eliminating loops. To evaluate the benefit of loop elimination, we attempt to artificially increase the number of existing loops in these multicast subgraphs. For this reason, we use a special topology, named Lattice-Topology, with 159 nodes and 284 links formed by joining two lattices. Each lattice contains 10×8 nodes and shares with the other lattice one border node. The source node is selected among the unshared border nodes of one lattice. Moreover, 25 destinations are randomly selected within the second lattice. Thus, we impose that all paths traverse the border node between the two lattices. This increases the occurrence of loops and enables to thoroughly evaluate the last measure of our study. The same process as Real-Topology is applied to generate the weights, the multicast groups and the requests.

Table I shows the number of detected and deleted loops by the greedy algorithm used in MAMCRA and our heuristics HCA and MLA. When the constraints are strict (area 1), most of the generated requests have few feasible paths. Therefore, few prefixes can be deleted without violating QoS constraints. In MAMCRA, the algorithm cannot delete more than 1.15 loops while it detects 13.42 loops. For HCA and MLA, the numbers of detected loops are 8 and 10.39 respectively while the numbers of deleted ones are 0.8 and 1.09 respectively. For loose constraints, the algorithm detects more loops but the number of deleted ones does not increase significantly.

We notice that the performances of HCA and MLA are close. Indeed, in these simulations we use non correlated weights. Thus, the shortest paths considering the hop count metric (with HCA) are very likely to correspond to the shortest paths minimizing the combined metric (with MLA).

[2]We notice by a prefix of a path the sub-path from the source node to an intermediate shared node between this path and another one

VI. Conclusion

The QoSMR problem is NP-Hard. After examining some of its earlier formulations, we define a more appropriate formulation that enables to evaluate the efficiency of the algorithms used to solve the QoSMR problem. We particularly propose two efficient heuristics that are compared to MAMCRA, which represents an efficient algorithmic solution. Taking into consideration the simulation environment, this comparison enables to draw four main conclusions. First, the success rates of the proposed heuristics are close to MAMCRA. Second, these heuristics are fast, especially for loose constraints where they become more than five times faster than MAMCRA. Third, the solutions computed by our heuristics are satisfying compared to MAMCRA, which in spite of its high computation time does not guarantee optimal solutions. Finally, the second step of MAMCRA also used in our heuristics, is not necessarily pertinent because of the small number of deleted loops. We conclude that the proposed heuristics are interesting to solve efficiently the QoSMR problem, with bounded combinatorial complexity while giving satisfying solutions.

As future work, we will attempt to substantiate the above cited conclusions by exploring larger environments with different topologies, using positively and negatively correlated weights. We will particularly study the impact of the multicast group size on our heuristics.

References

[1] H. Lin and Z. Yu-Lin and R. Yong-Hong, *Two multi-constrained multicast QoS routing algorithms*, eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007.

[2] T. Korkmaz and M. Krunz, *Multi-Constrained Optimal Path Selection*, IEEE INFOCOM, pp. 834-843, Anchorage, Alaska, 2001.

[3] G. Feng, *A multi-constrained multicast QoS routing algorithm*, Journal of Computer Communications, volume 29, pp. 1811-1822, 2006.

[4] H. Bettahar and A. Bouabdallah, *A new approach for delay-constrained multicast routing*, Journal of Computer Communications, volume 25, pp. 1751-1764, 2002.

[5] P. Khadivi, S. Samavi and T.D. Todd *Multi-constraint QoS routing using a new single mixed metrics*, J. Network and Computer Applications, 2008, volume 31, pp. 656-676.

[6] J. Y. Yen, *Finding the k shortest loopless paths in a network*. Management Science, volume 17, pp. 712-716,1971.

[7] F.A. Kuipers and P. Van Mieghem, *MAMCRA: a constrained-based multicast routing algorithm*, Computer Communications, volume 25, number 8, pp. 801-810, May 2002.

[8] P. Van Mieghem, H. De Neve and F.A. Kuipers, *Hop-by-hop Quality of Service Routing*, Computer Networks, volume 37, number 3-4, pp. 407-423, November 2001.

[9] P. Van Mieghem and F.A. Kuipers, *On the Complexity of QoS Routing*, Computer Communications, special issue on QofIS'01, volume 26, no. 4, pp. 376-387, March 2003.

[10] Z. Wang and J. Crowcroft,*Quality-of-Service Routing for Supporting Multimedia Applications*, IEEE Journal on Selected Areas in Communications, volume 14, pp. 1228-1234, 1996.

[11] J. Hershberger, M. Maxel, and S. Suri,*Finding the k shortest simple paths: A new algorithm and its implementation*, ACM Trans. Algorithms, volume 3, 2007.

[12] A. Zimolka, *Design of Survivable Optical Networks by Mathematical Optimization*, Technische University Berlin, 2007.

[13] H. Takahashi, A. Matsuyama, *An approximate solution for the Steiner problem in graphs*, Mathematica Japona, volume 24, pp. 573-577, 1980.