ILP formulation of the exact solution of multi-constrained minimum cost multicast

Walid Khallef, Sylvain Durand, Miklós Molnár

ILP Formulation of the Exact Solution of Multi-Constrained Minimum Cost Multicast

Walid Khallef[a,*], Sylvain Durand[b], Miklós Molnár[a]

[a] *University of Montpellier, LIRMM (UMR-CNRS 5506)*
161 rue Ada, 34095 Montpellier Cedex 5 France
[b] *University Paul-Valery, LIRMM (UMR-CNRS 5506)*
161 rue Ada, 34095 Montpellier Cedex 5 France

Abstract

Multimedia applications such as videoconferencing and collaborative applications require the satisfaction of several Quality of Service constraints (QoS). The routing with respect to QoS constraints was proposed in order to satisfy the user requirement and guarantee a certain level of performance to a data flow. As the communication architecture of these applications is often multicasting, the problem of finding a multicast route satisfying the QoS constraints proves to be challenging. In this paper we propose an Integer Linear Program (ILP) for finding the multicast route respecting a set of QoS constraints with minimum cost. Since the problem is NP-hard, we propose an efficient pretreatment algorithm (ArcReduce) to accelerate the resolution time. The pretreatment process can even answer in polynomial time, whether the problem has a solution or not, before starting the resolution process. The computation of the exact solution also allows for comparison of the heuristic solutions to the exact solution. We conduct an analysis of the ILP and the ArcReduce with various sizes of input data regarding the execution time, the success rate and the quality of the generated multicast route.

Keywords: Multicast routing, quality of service, multi-constrained Steiner problem, hierarchy, partial minimum spanning hierarchy.

1. Introduction

Multicast routing with Quality of Service (QoS) in networks is considered as an important field of research worldwide. With the appearance of next-generation multimedia Internet applications, much of work has been done on this subject to meet the requirements of users and to improve the communication in networks. The significant increase of connected users in Internet involves accessing large volume of data, often with QoS requirement which made these tasks more challenging [2][17].

*Corresponding author

Recently, with the advance in technology, multimedia applications are widely used, especially real-time multicast applications like videoconferencing, Video on Demand, Game on Demand, and Internet Protocol Television. In these applications the data are distributed from a source to several users. In Video on Demand, for instance, a single server delivers a high quality video to a large number of customers. Selecting a tree for routing, which is rooted at the source and contains all the destinations with minimal cost, is called the Steiner problem, which is NP-hard [3]. Exact algorithms as well as heuristics for solving the Steiner problem were proposed in the literature [16]. However, multimedia applications must respect several QoS constraints to operate properly (delay, delay jitter, bandwidth, packet loss...). Consequently, the basic algorithms designed for the Steiner problem are no longer adequate.

Therefore, several multicast routing algorithms are proposed not only for finding the multicast route which minimizes the cost, but also for satisfying the QoS constraints [5][6][8][11][10][17]. The problem of constructing a multi-constrained multicast route with minimal cost, the so-called Multi-Constrained Minimum Cost Multicast (MCMCM) problem is NP-hard.

In fact, there is a plethora of algorithms treating the MCMCM problem using different ways, but the goal is the same which is finding a multicast tree that respects a set of End-to-end constraints. Important works have adopted meta-heuristic techniques based on genetic algorithm [19][20], ant colony optimization [21][22] and tabu search algorithm [23].

The majority of MCMCM algorithms focusing on finding a multicast tree [11], while some other solve special cases of the problem (e.g. the Delay-Constrained Minimum Cost Multicast Routing Problem (DCMCM)) [10].

To solve MCMCM Kuipers et al. [8] proposed an efficient heuristic named MAMCRA. This algorithm computes a route which is not always a tree. First, MAMCRA calculates the set of optimal paths with minimum non-linear length[1] from the source to each destination using an exact multi-constrained path algorithm SAMCRA [14]. Since this set may contain some overlaps, MAMCRA uses in the second step a greedy algorithm to eliminate some of these overlaps without violating the constraints. The final structure is neither a tree nor an optimal solution.

In [11] Raayatpanah proposes a new QoS multicast solution following two steps. In the first step, data envelopment analysis (DEA) technique is used to evaluate the arc efficiency in the network. Afterward, each arc in the network can be considered as a decision making unit by replacing its weights to be minimized and maximized with inputs and outputs respectively. In the second step, the problem is formulated as an integer linear programming based on the relative efficiency scores of arcs to determine the multicast tree.

Actually the problem of QoS multicast routing is a hot topic in wireless networks and related applications. Since the problem is NP-hard, Lu and Zhu

[1]The non-linear length function permits to normalize the constraints and take into account the most critical one of a path which is the non-linear-length of this path.

proposed in [26] a genetic algorithm based and energy efficient heuristic for QoS multicasting in MANETs.

Li Wei et al. [24] designed the (EA-ACA-QMRA) algorithm for MCMCM problem which is the combination of an evolutionary algorithm (EA) and an ant colony algorithm (ACA). The key potential of EA-ACA-QMR is the rapid global search capability of EAs with the pheromone feedback factor of ACAs while accounting for multiple constraints. In short, the algorithm combines EA and ACA to describe a strategy for satisfying multiple QoS constraints on a multicast tree.

An interesting algorithm has proposed in [27] to optimize the multi constrained multicast tree using Teaching Learning Based Optimization method (TLBO). TLBO is a population-based method, the population here is considered as a class of learners. The TLBO process is divided into two parts. The first part is the Teacher phase which means learning from teacher and the second part is Learning phase which means learning by the interaction between learners. The TLBO method works on the effect of influence of a teacher on learners in a class. So, output is considered in term of results or grades. Naik et al. have used this concept of algorithm for optimization MCMCM problem. According to the authors, in term of optimal solution quality TLBO performs better than the existing MCMCM algorithms.

Ajay et al. [25] proposed a new method based on fuzzy logic. In this mechanism all the metrics of the routs are combined into the same metric (i.e. fuzzy metric). After though, the minimum fuzzy cost solution will be considered as the optimal solution. The proposed fuzzy logic is very interesting when the goal is to optimize globally the network. In other words, when there are no defined constraints to respect. Unfortunately, it is not the case with some recent applications having strict requirements (e.g. the end-to-end delay must be less than 30 ms, etc.). Therefore, using the combination of metrics does not guarantee that each path respects the end-to-end constraints from the root to any node.

It will be interesting to manage the problem of QoS routing with a general framework with resource allocation [28]. The crux of the method is the formulation of QoS routing, which incorporates the hardware/software implementation and its relation to the allocated resources into a single framework. Our biggest concerns in this paper is that of finding a multicast QoS route that satisfies the end-to-end constraints without resource allocation.

As mentioned in [15] the optimal solution of MCMCM is not a tree (not a sub-graph) but a hierarchy (cf. Section 2). To the best of our knowledge, the exact ILP formulation has not yet been published. The first part of the paper presents the proposed Integer Linear Program (ILP) that finds optimal hierarchies for MCMCM. It is based on a multi-flow method. The ILP gives the optimal route, if it exists. This ILP can also be used to evaluate the efficiency of earlier proposed heuristics. In this paper we propose tests for MAMCRA, which is one of the most efficient heuristics to date. The second part of the paper presents, a new pretreatment algorithm designed to reduce the search area. The pretreatment step consists of eliminating the arcs that cannot be part of the feasible QoS multicast route. Therefore, the pretreatment algorithm

accelerates the process and improve significantly the scalability of the ILP.

The rest of the paper is organized as follows. Section 2 presents the problem formulation based on the hierarchies. We demonstrate in the same section some properties of hierarchies and the MCMCM problem. In Section 3 we illustrate the limitation of using standard ILP formulations. Section 4 describes the ILP and explains why we propose a multi-flow method. The efficiency of our ILP is demonstrated in Section 5. Section 6 presents the pretreatment algorithm called ArcReduce, while Section 7 is the experimental part of ArcReduce. Finally, Section 8 concludes this paper.

2. Problem formulation

2.1. Hierarchies

In order to define the optimal solution, we recall in this section the concept of hierarchies accompanied by an illustrative example.

Definition 1. (Hierarchy) Let $T = (W, F)$ be a tree and $G = (V, E)$ a connected graph. A homomorphism $h : W \longrightarrow V$ maps each vertex in W to a vertex in V such that the mapping preserves the adjacency, i.e., $(u, v) \in F \Longrightarrow (h(u), h(v)) \in E$. The triple (T, h, G) defines a **hierarchy** H in G. The resulting hierarchy can also be represented as $H = (V', E')$, where V' and E' are multi-sets.

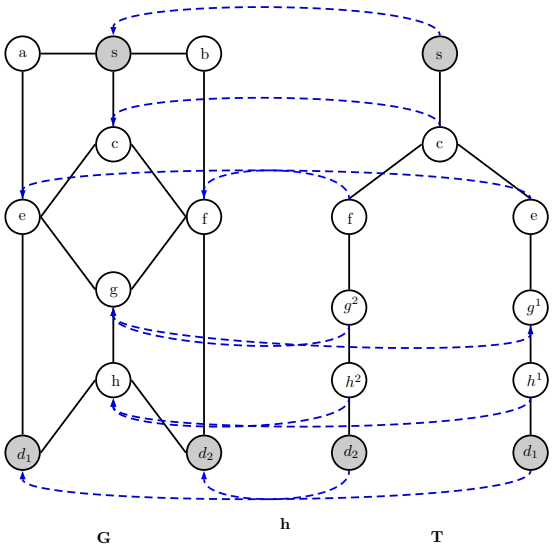

Figure 1: Mapping of vertices for a hierarchy

Figure 1 gives an example applying a mapping h from a tree T to a target graph G for a hierarchy H. Each vertex of T is associated with a unique vertex

of G. In the reverse direction, some vertices of G are not concerned by the mapping and some others mapped more than once in T, e.g., vertices e and f. As we can see, the set of vertices and edges induced by T in G is not a subgraph of G. It remains connected and can be used as a spanning structure for multicast routing. Especially when the mapping h is injective, the hierarchy corresponds to a tree. Thus, a tree is a special case of hierarchy. To distinguish the occurrences in hierarchy H associated with the same vertex v in G, we label them $v^1, v^2, \ldots, v^k$ in T. The concept of hierarchy can naturally be extended to directed graphs.

2.2. Model and notation

A communication network may be modelled as an undirected weighted graph $G_n = (V, E)$, where V is the set of vertices and E the set of edges. In our problem each edge $e \in E$ is associated with M weights given by a weight vector $\vec{w}(e) = [w_1(e), w_2(e), ..., w_M(e)]^T$ representing the QoS weights, and with $c(e)$ representing the cost for using the edge. 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. Here, we only consider additive metrics. The multiplicative metrics can be transformed into additive metrics by using a logarithm function. Bottleneck metrics are beyond the scope of this paper. However, we can easily consider bottleneck metrics (e.g. bandwidth) adding a new constraint in our ILP (cf. Section 4).

The end-to-end requirement based on additive metrics is expressed as constraints from the source to the destinations. It is given by an M-dimensional constraint vector $\vec{L} = [L_1, L_2, ..., L_M]^T$. To simplify, we suppose that the same constraints are imposed for all destinations. For additive metrics, the weight of a path $p(s, d_j)$ corresponding to the metric i is given by:

$$l_i(p(s, d_j)) = \sum_{e \in p(s,d)} w_i(e). \tag{1}$$

Our problem aims to find the route $H^* = (V^*, E^*)$ rooted at the source vertex s and spanning a non-empty destination set $D = \{d_1, d_2, ..., d_d\}$, respecting the end-to-end requirement from the source to each destination (i.e. containing a path from the source to each destination). The cost of the route $c(H^*)$ is the sum of the cost of all edges in H^*. Furthermore, the cost of an edge $e \in H^*$ can multiply several times to this sum if it is crossed by several paths Figure 1 edge (g, h).

Definition 2. (MCMCM problem) For s, D given, find $H^* = (V^*, E^*)$ containing for each destination $d_j \in D$ a path $p(s, d_j)$ between s and d_j, such that

$$l_i(p(s, d_j)) \leq L_i, i = 1, \ldots, M, j = 1, \ldots, d \tag{2}$$

and the cost $c(H^*)$ is minimum.

Notice: In $H^* = (V^*, E^*)$, V^* is a set of vertex occurrences and E^* the set of edge occurrences. The route may refer to vertices/edges several times (it is not a sub-graph but a spanning hierarchy). Consequently, one vertex or one edge may be present several times in V^* and E^* respectively.

Property 1. *MCMCM is NP-hard for $M \geq 2$.*

Proof. Let us consider MCMCM. For $M \geq 2$, $|D| = 1$ is the multi-constrained optimal path problem (MCOP), which is proved to be NP-hard [7]. □

2.3. Properties of the solution of MCMCM

Property 2. *The solution H^* of the MCMCM is not necessarily a tree.*

Proof. As we show in Figure 2.A, in the optimal spanning solution, a vertex or an edge can be crossed several times.
In this example, there are two possible paths to reach each destination in $\{d_1, d_2\}$. $(s-a-b-e-f-d_1)$ and $(s-a-c-e-f-d_1)$ to reach d_1 with a total weight of $[9,9]^T$ and $[10,8]^T$ respectively, $(s-a-b-e-f-d_2)$ and $(s-a-c-e-f-d_2)$ to reach d_2 with a total weight of $[10,8]^T$ and $[9,9]^T$ respectively. If the end-to-end QoS constraints are given by $\vec{L} = [9,9]^T$, the only feasible paths for d_1 and d_2 are $(s-a-c-e-f-d_1)$ and $(s-a-b-e-f-d_2)$ respectively. The cost optimal solution is not the set of the two paths. Messages from s may be sent only once, the edge (s,a) can be shared for the two destinations and the messages should be duplicated in vertex a. After the duplication, the messages follow the remaining route to the destinations. It is clear that the route generated by these two paths is not a tree in G. □

Property 3. *Even when there is a feasible tree, a hierarchy satisfying the constraints may be cheaper.*

Proof. As shown in Figure2, we can reach each destination in $\{d_1, d_2\}$ using the hierarchy or a tree $((s,(a,(b,(e,(f,d_2)))),(c,d_1))))$. However, in terms of cost, the hierarchy cost of 9\$ (the cost of (e,f) in the hierarchy solution is counted twice since the two paths do not have a common prefix to reach vertex e), while the tree cost is 12\$. □

Theorem 1. *The optimal multicast route H^* with respect to multiple constraints on positive additive metrics is always a directed partial spanning hierarchy, if it exists [12].*

Definition 3. (Shared arcs) In the optimal hierarchy H^*, two paths or more can share the same arc occurrence if they have a common prefix[2]. On the contrary, an arc must be used several times if it is crossed by two or more paths and no common prefix exist between these paths.

[2]The common prefix between two paths P and P' corresponds to the set of links from the source node to the furthest common node without bifurcation between these two paths.

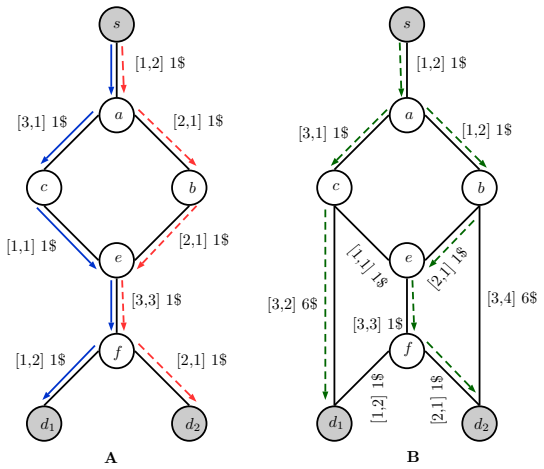

Figure 2: Example of the optimal hierarchy for $\vec{L} = [9, 9]^T$

3. Limitation of standard ILP formulations

In the classical formulation of the MCMCM problem [8] the goal is to find a subgraph G' of G having a feasible path from the source s to each destination $d_j \in D$, and the cost of the sub-graph is minimal. This definition can be used to formulate the MCMCM problem by an integer program which introduces some flow variables and special constraints to ensure the feasibility of each path [11]. Using the same definition, the classical cut formulation of spanning trees can also be used. However, in [12], Molnar et al proved that hierarchies solve MCMCM and they are not sub-graph. Since a hierarchy can return to vertices and cross arcs several times, flow- and cut-based formulations in the topology graph are not possible (usual flow conservation conditions and cut formulations cannot be expressed). Thus, the classical formulations are no longer adequate. We propose a special flow-based method in a particular undirected multi-graph to solve the problem for the following reasons:

- In the worst case an edge can be used $|D| = d$ times in a given direction, if all paths of the multicast route use it and no common prefix exists between them. An example is the edge (e,f) in Figure 3 which can easily be generalized. Hence we duplicate each initial edge d times in both directions to build a directed multi-graph (digraph). This separation of the edge/arc occurrences permits the coding of the flows traversing the original edges several times and in both directions.

- Two or more flows can share the same arc e in the digraph if, and only if, the prefix of these paths to the arc e is the same. In Figure 3, p_1 and p_2 share the arc (s, a) because $pref(p_1, (s, a)) = pref(p_2, (s, a))$, but they do not share the arc (e, f) because $pref(p_1, (e, f)) \neq pref(p_2, (e, f))$. A simple flow method cannot ensure this property.

- Vertices are not duplicated because the flow conditions can be expressed even if several flows traverse some vertices.

4. ILP Formulation

In this section we propose an Integer Linear Program formulation of the hierarchies solving the MCMCM problem.

We define a multi-graph $G = (V, A)$ obtained by the duplication of each edge of the topology graph of the network $d = |D|$ times in both directions: $|A| = 2d * |E|$.
We explain below the network parameters in Table 1 and the variables used to realize this ILP in Table 2.

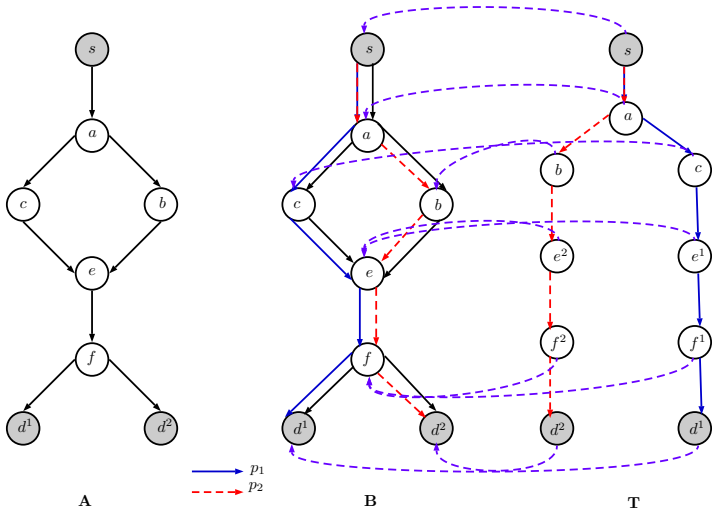

Figure 3: (A) initial graph; (B) multi-graph; (T) routing solution.

Objective

The objective function of minimizing the total cost of the solution can be expressed as follows:

$$\text{minimize} \sum_{m \in V} \sum_{n \in Out(m)} \sum_{i \in \{1,\dots,d\}} U_i(m,n) \cdot c(m,n) \tag{3}$$

Constraints

- Paths constraints:
For the source

$$\sum_{n \in Out(s)} \sum_{i \in \{1,\dots,d\}} \sum_{k \in \{1,\dots,d\}} B_i(k,s,n) = d \tag{4}$$

Table 1: Network parameters

Parameter	Description
$G = (V, A)$	The digraph obtained by the duplication of edges from the network topology (each edges of the original network is present $d = \|D\|$ times in both directions).
s	Source.
$D = \{d_1, d_2, ..., d_d\}$	Set of destinations.
$In(m)$	Set of vertices leading an arc to vertex m.
$Out(m)$	Set of vertices forwarded by arcs from m.
$\vec{w}(m,n) =$ $[w_1(m,n), ..., w_M(m,n)]^T$	Weight vector of arc(m,n).
$c(m,n)$	Cost of arc(m,n).
$\vec{L} = [L_1, L_2, ..., L_M]^T$	Constraint vector for each destination.

Table 2: ILP Variables

Variable	Description
$U_i(m,n)$	Binary variable. Equals to 1 if the multicast route uses the i^{th} duplication of arc (m,n), equals to 0 otherwise ($d \cdot \|A\|$ variables).
$B_i(k,m,n)$	Binary variable. Equals to 1 if path $p(s,d_k)$ in the solution uses the i^{th} duplication of arc (m,n), equals to 0 otherwise ($d^2 \cdot \|A\|$ variables).
$R_i(k,l,m,n)$	Binary variable. Equals to 1 if paths $p(s,d_k)$ and $p(s,d_l)$ in the solution share the same i^{th} duplication of arc (m,n), equals to 0 otherwise ($d^3 \cdot \|A\|$ variables).

For the destinations

$$\forall d_k \in D$$
$$\sum_{m \in In(d_k)} \sum_{i \in \{1,...,d\}} B_i(k, m, d_k) - \sum_{n \in Out(d_k)} \sum_{i \in \{1,...,d\}} B_i(k, d_k, n) = 1 \quad (5)$$

For the other vertices

$$\forall m \in V \setminus \{D \cup \{s\}\}, \forall k \in \{1, ..., d\}$$
$$\sum_{q \in In(m)} \sum_{i \in \{1,...,d\}} B_i(k, q, m) = \sum_{n \in Out(m)} \sum_{i \in \{1,...,d\}} B_i(k, m, n) \quad (6)$$

The constraints 4, 5 and 6 guarantee (in this order) that the number of outgoing paths from the source equals the number of destinations, each destination vertex has exactly one path from the source. The incoming flow in the intermediate vertices equals the outgoing (conservation of flow).

- QoS constraints:

$$\forall k \in \{1, ..., d\}, \forall h \in \{1, ..., M\},$$
$$\sum_{m \in V} \sum_{n \in Out(m)} \sum_{i \in \{1,...,d\}} B_i(k, m, n) \cdot w_h(m, n) \leq L_h \quad (7)$$

The constraint 7 guarantees that each path respect the given QoS constraints.

- Constraints for linking paths and usage of arcs:

$$\forall (m, n) \in A, \forall i \in \{1, ..., d\} ,$$
$$U_i(m, n) \leq \sum_{k \in \{1,...,d\}} B_i(k, m, n) \quad (8)$$

$$\forall (m, n) \in A, \forall i, k \in \{1, ..., d\} \times \{1, ..., d\} ,$$
$$U_i(m, n) \geq B_i(k, m, n) \quad (9)$$

Constraints 8 and 9 ensure that each arc in the multicast route is traversed by one path at least or by all paths at most, i.e. the multicast route contains only the set of arcs used in the solution.

- Constraints for setting the values of $R_i(k,l,m,n)$:
Remember these variables indicate if an arc occurrence is used commonly by two paths. Trivially $R_i(k,l,m,n) = B_i(k,m,n) \cdot B_i(l,m,n)$ and this condition can be linearized as

$$\begin{aligned} \forall\{k,l\} \in \{1,...,d\} \times \{1,...,d\}, \forall (m,n) \in A, \forall i \in \{1,...,d\}, \\ R_i(k,l,m,n) \leq B_i(k,m,n) \\ R_i(k,l,m,n) \leq B_i(l,m,n) \\ R_i(k,l,m,n) \geq B_i(k,m,n) + B_i(l,m,n) - 1 \end{aligned} \tag{10}$$

The following constraints force paths not to share arcs if it is not already the case in a the previous arcs

$$\begin{aligned} \forall\{k,l\} \in \{1,...,d\} \times \{1,...,d\}, \forall m \in V, V \backslash \{s\}, \\ \sum_{i \in \{1,...,d\}} \sum_{q \in In(m)} R_i(k,l,q,m) \geq \sum_{i \in \{1,...,d\}} \sum_{n \in Out(m)} R_i(k,l,m,n) \end{aligned} \tag{11}$$

Constraints 11 ensure that an arc (m,n) is shared by two paths p_1 and p_2, if and only if it is already (recursively) the case in a previous arcs (q,m) (there is a common prefix for the two paths).

Property 4. *The optimal solution of the MCMCM problem computed by the ILP in the multi-graph corresponds exactly to the optimal hierarchy to solve the MCMCM in the original graph G_n.*

Proof. Let us suppose that $H_1 = (T, h_1, G)$ is an optimal solution in G computed by the ILP. Using the mapping between the arcs of G and the edges of G, we obtain a hierarchy $H_2 = (T, h_2, G_n)$ in G_n. Each path $p(s, d_i), \forall d_i \in D$ corresponds to the QoS constraints (by construction of H_1). We should prove that H_2 is optimal in G_n.
Proof by contradiction: Let us suppose that H_2 is not optimal and there is a hierarchy H_3 corresponding to the QoS constraints in G_n s.t. $c(H_3) < c(H_2)$. Let H_4 be the hierarchy projected in G following H_3. H_4 corresponds to the QoS constraints and $c(H_4) < c(H_1)$ which is in contradiction with the fact that H_1 is of minimum cost. □

Example: Figure 4.A presents the original graph G_n when the objective is finding the optimal solution of MCMCM from the source s to the destination set $D = \{d_1, d_2, d_3\}$. Figure 4.B presents the multigraph $G = (V, A)$ obtained by the duplication three times of each edge in the original graph (i.e. $|D| = 3$) in both directions. G_n and G are defined on the same vertex set. Each edge in E corresponds to $d * |E|$ arcs in A in each direction.

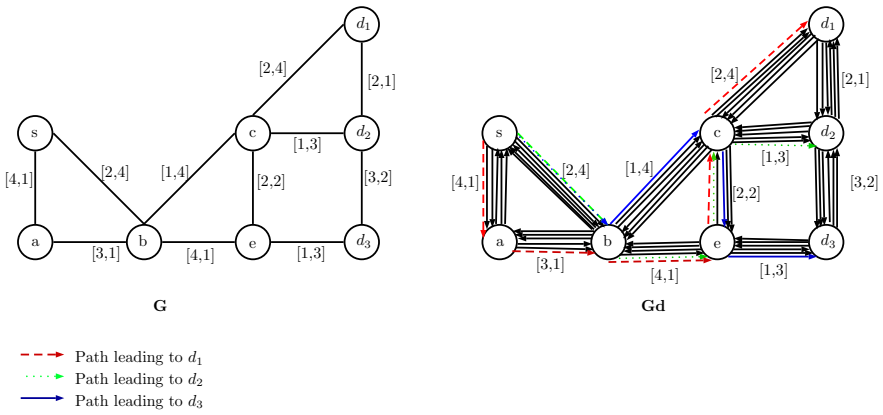

Figure 4: The original graph and the multigraph

As mentioned before, our ILP model doesn't take into account the bottleneck metrics. To adapt our ILP to be able to manage the bottleneck constraint we have to add an additional constraint (Constraint 12), which ignores the usage of edges that doesn't satisfy the critical corresponding bottleneck value during the calculation of the route. For example, we can limit the usage of edges that don't respect the bandwidth constraint by adding the following constraint to our ILP.
Let $Bb(m,n)$ be the bandwidth capacity of the edge (m,n). Let $b_i(m,n)$ be the debit of bandwidth used when the route crosses the i^{th} duplication of edge (m,n).

$$\forall (m,n) \in E,$$
$$\sum_{i \in \{1,\ldots,d\}} U_i(m,n) + \sum_{i \in \{1,\ldots,d\}} U_i(n,m)) \cdot b_i(m,n) \leq Bb(m,n) \quad (12)$$

Notice, usually the bandwidth used by a multicast communication is the same for all edge/arc occurrences ($b_i(m,n) = bw, \forall (m,n) \in A, i = 1, \ldots d,$).

5. Experiments

The objective of our experimentation is to analyze the optimal routes which correspond to hierarchies and to compare the solutions of a very known heuristic MAMCRA to these optimums.

Datasets. We evaluate the practical potential of our ILP on random graphs generated by the WAXMAN model [13], and also on two well known typical networks: on the NSF topology, and on the NTT topology [14].
Remember, in the WAXMAN model, the vertices are randomly placed in a unit square. The probability of creating an arc between two vertices i and j is

$\beta \exp(\frac{-d_{ij}}{\alpha L})$, where $0 < \alpha, \beta \leq 1, d_{ij}$ is the Euclidean distance from i to j and L is the maximum distance between two nodes.

Each arc of the graphs is weighted by integer metrics, representing the QoS metrics and cost. These values of arc are randomly chosen in {1,..,10}.
We also analyze the efficiency of the MAMCRA algorithm using the same topologies.

Each experimentation in this section was conducted on 50 instances. The type of networks, the number of vertices and arcs are presented in Table 3.

Table 3: Dataset information

Networks	Information			
	$\|V\|$	$\|A\|$	α	β
NSF topology	14	21	-	-
NTT topology	55	144	-	-
WAXMAN 50	50	300	1	0.11
WAXMAN 100	100	600	1	0.073
WAXMAN 200	200	1200	1	0.048

Since each link in real networks is bidirectional, we organize the run of experiments in the following manner. Each link is replaced by two opposite arcs asymmetrically weighted. The values of the QoS metrics and the cost may be different in each direction.

Main parameters for the performance analysis. We study the behavior of our ILP and MAMCRA algorithm regarding two main parameters.
- *Destinations vertices Density (DD):* Percentage of destinations $D \cup \{s\}$ regarding the total number of vertices $|V|$.
- *Constraints Looseness (CL):* The CL is the average value of $\frac{L_i}{max - value_i}, i \in \{1, \ldots, M\}$, where $max - value_i$ is the maximum possible value of metric of index i. Since the values of the QoS metrics for each link are in {1,...,10}, $max - value_i = 10$ for all metrics. In order to generate instances with a specific CL, all L_i are equal to 10*CL.

We fixed the number of destinations to 5 and the CL to 6 when the DD (or the number of destinations) and CL are not mentioned. The number of constraints is set to 2 if it is not mentioned.

Environment. The experiments are made on a 2 cores PC (Intel i3-3110M) having 4GB of memory and running under Windows 7 (64bit). The ILP is solved with IBM ILOG CPLEX 12.6.2 [3].

[3] http://www-03.ibm.com/software/products/en/ibmilogcpleoptistud/

Table 4: CPU time

Datasets	CPU time				
Number of destinations	2	4	6	8	10
NSF topology	0.07s	0.31s	1.05s	4.14s	7.45s
NTT network	0.09s	3.10s	5.02s	19.40s	28.80s
WAXMAN 50	0.11s	5.18s	8.67s	28.09s	47.01s
WAXMAN 100	0.66s	20.18s	125.90s	380.30s	680.40s
WAXMAN 200	1.95s	90.18s	522.90s	725.30s	DNF

5.1. Execution time

First, we investigate the CPU time to solve our ILP, since the execution time plays an important role in network routing. We run our ILP with different network topologies, Table 4 presents the average CPU time with different destination sets. DNF means it did not finish in one hour. In small networks and for small group sizes (e.g. 2 destinations and 50 nodes), the execution time is found to be reasonable. These values permit the application of the exact computation in small Software Defined Network [18] for off-line route computation. As we show, the larger the number of destinations $|D|$, the more complex the solution will become, due to the way that the edges of the original graph are duplicated which influences considerably the CPU time, hence the algorithm will compare more combinations to find the best one.

Table 5: Proportion of hierarchies in feasible instances

Number of destinations	5 destinations			
Constraints looseness	CL = 6		CL = 7	
Number of constraints	FI	POH	FI	POH
2	60.60%	29.83%	61.00%	27.72%
4	58.40%	28.23%	58.80%	25.34%
6	57.20%	34.61%	57.40%	34.49%
8	54.80%	33.94%	56.00%	32.14%

Number of destinations	8 destinations			
Constraints looseness	CL = 6		CL = 7	
Number of constraints	FI	POH	FI	POH
2	20.40%	64.70%	20.40%	55.88%
4	16.00%	58.75%	18.00%	46.66%
6	15.60%	62.82%	17.40%	51.72%
8	14.60%	63.01%	16.40%	51.21%

5.2. Proportion of optimal hierarchies

We investigate the proportion of instances that do not admit a tree as a solution, but a hierarchy. We conduct this test on 500 instances in the NTT topology. The set of destinations and the source are randomly chosen. Table 5 presents the proportion of Feasible Instances (FI) and among them, the percentage of the optimal solutions which are hierarchies and not trees (POH).

This test demonstrates the interest of the hierarchy and shows the frequency of this phenomenon in real networks. As shown in Table 5, the rate of instances for which the optimal solution is a hierarchy and not a tree is strongly correlated with:

Number of destinations: When the number of destinations increases the number of paths increases as well, which causes more crosses between paths to obtain the exact solution.

Number of constraints: With a large number of constraints the matching between paths will be less probable and the generation of overlaps between paths increases which explains the obtained results.

Looseness of constraints: The numerical results in Table 5 show that the proportion of hierarchy decreases when the constraints are more loose ($CL = 7$). However, it is demonstrated that even when the constraints are loose, the hierarchy may be an optimal solution (Property 3). According to the results, the cases whose optimal solution are represented by a hierarchy are very rare when the constraints are loose. With loose constraints the solution converges to the Steiner tree.

5.3. Evaluation of MAMCRA algorithm

In the following tests, we implement two different versions of MAMCRA. MAMCRA-NLF with the Non-linear Length Function (NLF) is proposed in [5]. It minimizes this non-linear length function without taking into account the optimization of the cost, while MAMCRA-cost tries to optimise the cost.

We investigate in each experimentation the cost of the multicast route.

CL tests: Cost of the multicast structure generated regarding the constraint's looseness. Figure 5, shows that the solutions given by MAMCRA-NLF become worse when the constraints are very loose. The quality improves when the constraints are tightened. The main reason for this may be that when the number of feasible paths is small, MAMCRA-NLF and the ILP frequently choose the same paths. On the contrary, the quality of the solution returned by MAMCRA-cost is very good, especially when the constraints are very loose.

DD tests: Cost of the multicast structure generated regarding destinations vertices density. Figure 6, shows that the solutions given by MAMCRA-NLF are further from the optimum when the DD increases, because the number of edges that construct the multicast route are increasing to cover all destinations. Meanwhile, MAMCRA-NLF chooses the feasible edges without optimizing the cost.

Here again, the solution returned by MAMCRA-cost are very close to the optimum. In terms of complexity MAMCRA-cost takes more time than MAMCRA-NLF to find the solution, however MAMCRA-cost gives an efficient (even if not

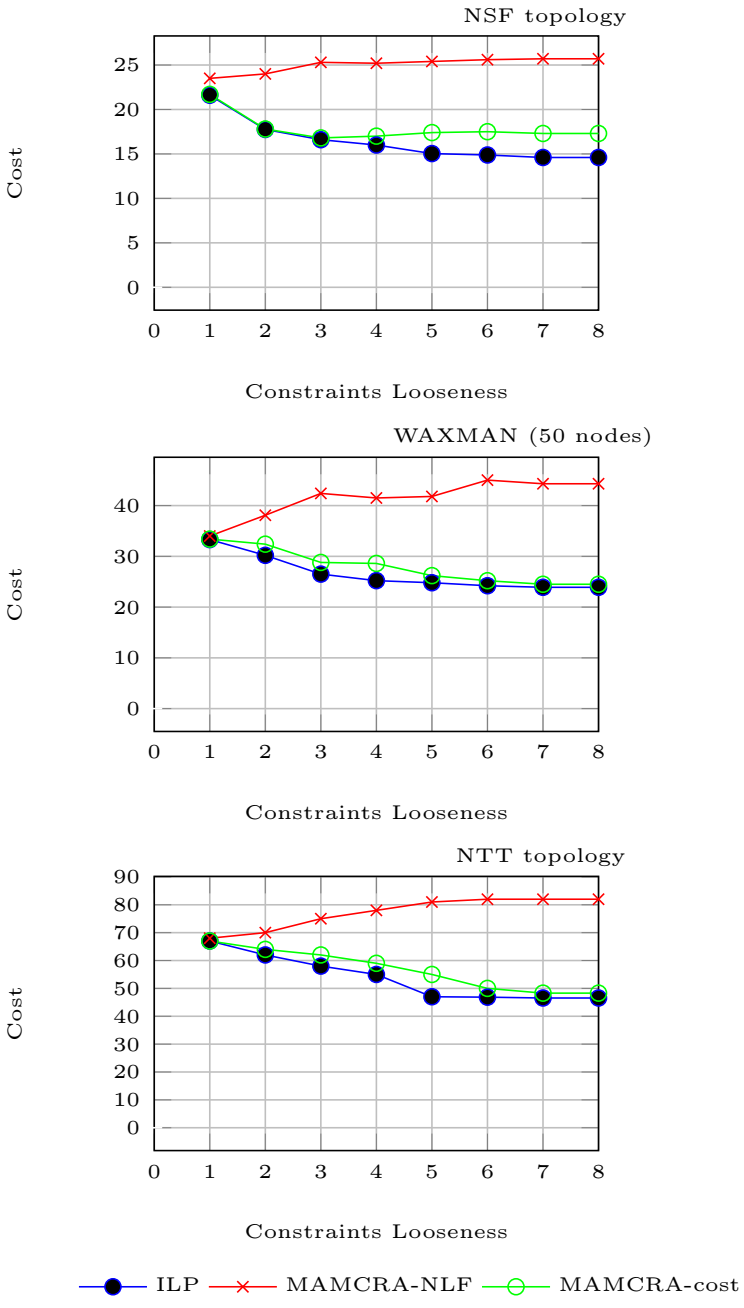

Figure 5: CL tests: Cost regarding to the Constraints Looseness.

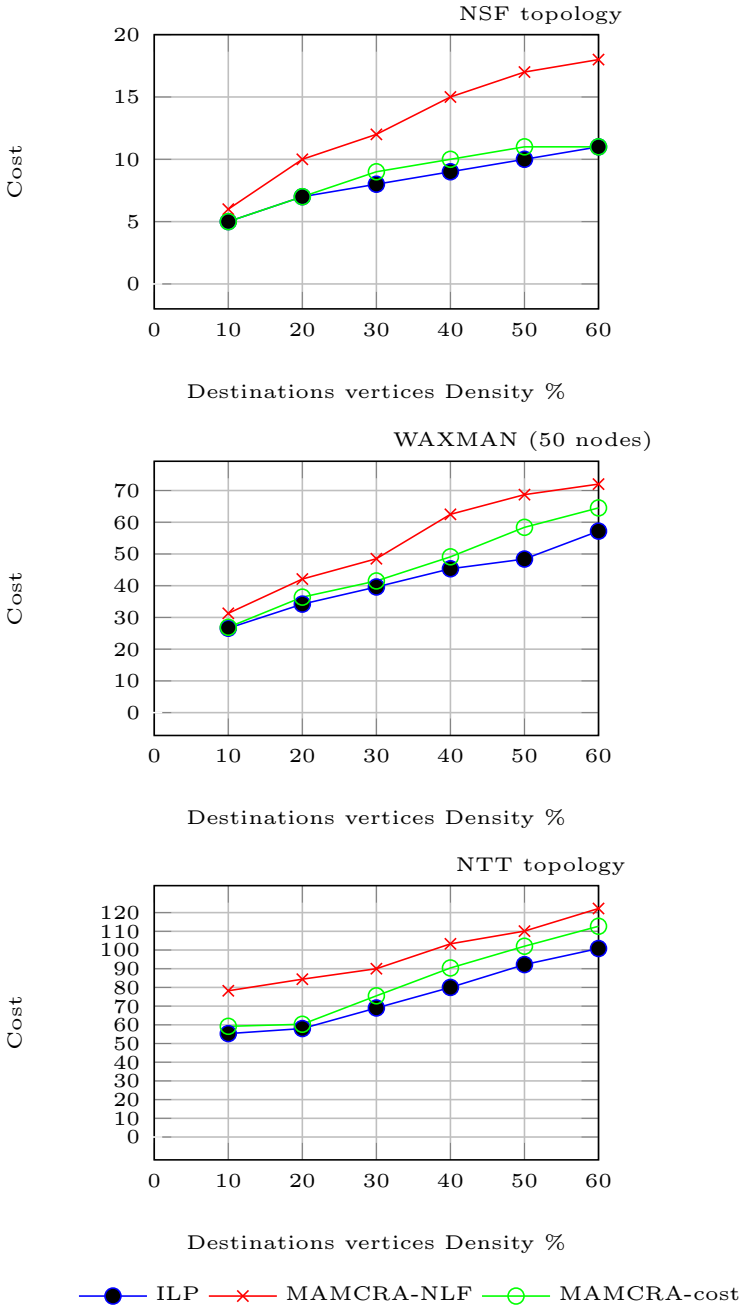

Figure 6: DD tests: Cost regarding the Destinations vertices Density.

always optimal) solution.

6. Pretreatment ArcReduce

To solve the QoS routing more efficiently we propose a new algorithm ArcReduce which prunes the weighted topology graph to produce a reduced graph with fewer arcs.

6.1. Motivation

Usually the complexity of all proposed QoS routing algorithms and our ILP is strongly correlated with the number of arcs/vertices. Therefore, the size of the input data graph is fundamental. The role of ArcReduce is to eliminate the arcs that cannot be part of the feasible QoS multicast routes.

Definition 4. Let $(u, v) \in A$ be an arc. Let $l_i^*(s, u)$ and $l_i^*(v, d)$ be the length of the shortest path from the source s to u and from v to the destination d respectively concerning the metric $i \in M$.
The arc (u, v) is **useless** (i.e. cannot be used in a route from s to d respecting the QoS constraints) if:

$$l_i^*(u, v) + w_i(u, v) + l_i^*(v, d) > L_i \tag{13}$$

Property 5. *An arc (u, v) can be eliminated from the graph G if it is useless for all destinations in D.*

Proof. The multicast route necessarily has a path from the source to each destination $p(s, d_j), d_j \in D$. If we have an arc a that is feasible for only one destination, we cannot drop it because it may be used to construct the feasible path to this destination. □

Theorem 2. *Let (x, y) be useless arc for G. An arc (x', y') is useless for G if and only if it is useless for $G\backslash\{(x, y)\}$.*

Proof. Let (x, y) be useless arc for G. $l_i^*(a, b)$ is the length of a path $p(a, b)$ using the weight of index i So:

$$\forall j \in \{1...d\},\ \exists i \in \{1...M\} \mid l_i^*(x, y) + w_i(x, y) + l_i^*(y, d_j) > L_i \tag{A}$$

- If (x', y') is useless for G, it is obviously useless for $G\backslash\{(x, y)\}$.
- Suppose that (x', y') is useless for $G\backslash\{(x, y)\}$ and not for G.

$$\text{Then } \exists j \in \{1...d\} \mid \forall i \in \{1...M\},\ l_i^*(s, y') + w_i(x', y') + l_i^*(y', d_j) \leq L_i \tag{B}$$

Since this property is false in $G\backslash\{(x, y)\}$, it means that:
$\exists i \in \{1...M\}$ — $(x, y) \in l_i^*(s, x')$ or $(x, y) \in l_i^*(y', d_j)$
If $(x, y) \in l_i^*(s, x')$ then (B) implies that, $l_i^*(s, x)+w_i(x, y)+l_i^*(y, x')+w_i(x', y')+l_i^*(y', d_j) \leq L_i$.
But $l_i^*(y, x')+w_i(x', y')+l_i^*(y', d_j) \geq l_i^*(y, d_j)$ so $l_i^*(s, x)+w_i(x, y)+l_i^*(y, d_j) \leq L_i$ which contradicts (A).
The argumentation is the same if $(x, y) \in l_i^*(y', d_j)$. □

6.2. The ArcReduce algorithm

The meta-code of the ArcReduce algorithm is listed below:

Algorithm 1 ArcReduce

Require: $G(V, A)$: digraph, s : source, $D = \{d_1, \ldots, d_k\}$: destinations set, $\overrightarrow{L} = [L_1, \ldots, L_M]^T$: constraints.
Ensure: $G'(V', A')$: reduced graph.

```
1:  ArcEliminated := ∅;
2:  for all (u, v) ∈ A do
3:     InfeasibleDestination:= 0;
4:     for all d ∈ D do
5:        InfeasibleConstraints:= 0;
6:        i = 0;
7:        while (i ≤ M and InfeasibleConstraints := 0) do
8:           i = i + 1;
9:           if l_i^*(u, v) + w_i(u, v) + l_i^*(v, d) ≥ L_i then
10:             InfeasibleConstraints:=InfeasibleConstraints+1;
11:          end if
12:       end while
13:       if FeasibleConstraints ≠ 0 then
14:          InfeasibleDestination := InfeasibleDestination+1;
15:       end if
16:    end for
17:    if InfeasibleDestination == |D| then
18:       Add (u, v) to ArcEliminated ;
19:    end if
20: end for
21: G*(V*, A*) == G(V, A\ ArcEliminated) ;
```

For each $(u, v) \in A$, ArcReduce checks the usefulness. The algorithm calculates for each metric $i \in \{1, \ldots, M\}$ the shortest path l_i^* from the source to (u) and from (v) to the destination, using Dijkstra's algorithm and check the feasibility following Definition 4 (lines 6-9). If the arc (u, v) is useless for one constraint the algorithm does not have to check the other constraints and repeat the same calculation with all $d \in D$. If the arc (u, v) is feasible for one destination $d \in D$ the algorithm does not have to check the other destinations. Otherwise, if the arc is useless for all destinations $d \in D$ (Property 5), the arc will be dropped from A.

6.3. Complexity

ArcReduce calculates the shortest path using Dijkstra's algorithm. The complexity of Dijkstra's algorithm using a Fibonacci heap is $O(|A|+|V|\log|V|)$. For all arcs the complexity is $O(|A|(|A|+|V|\log|V|)$. The for loop starting on line

4 is invoked at most $O(|D|)$ times. Calculating the feasibility of the arc with all constraints line 6 takes at most $O(M)$ times. The worst-case time-complexity of ArcReduce is:

$$O((|A||D|M)(|A| + |V| \log |V|)). \tag{14}$$

Remember that M is the number of QoS metrics and $|D|$ is the number of destinations.

7. Experiments of ArcReduce

7.1. Percentage of eliminated arcs

We investigate the percentage of the eliminated arcs regarding the three performance parameters used in the experiments of the ILP: *Number of Destinations* (ND), *Constraints Looseness* (CL) and *Number of Constraints* (NC).

Table 6: Percentage of eliminated arcs with ArcReduce (100 nodes)

ND	2 destinations			4 destinations		
NC	2	4	6	2	4	6
CL= 2	90.82%	91.12%	91.41%	87.69%	94.03%	94.93%
CL= 3	53.61%	81.96%	86.24%	36.44%	64.67%	68.52%
CL= 4	5.29%	16.34%	17.82%	1.82%	5.29%	6.40%

7.2. Execution time of the ILP with and without using ArcReduce

Figure 7 represents the comparison between the runtime of the three algorithms (ILP with/without using ArcReduce and ArcReduce). The CL and NC values are fixed in 3 and 2 respectively. As shown by the results above, we can gain important time using the ArcReduce. In turn, the ArcReduce filters the graph in a negligible amount of time compared the execution time of the ILP (Figure 7).

7.3. Discussion

It is clear that the percentage of the eliminated arcs depends strongly on the CL, view Table 6. ArcReduce is effective when the problems become harder (i.e. the constraints are very strict) because the number of eliminated arcs is very significant, which significantly accelerates the resolution process (cf. Figure 7). In cases where the problem is easier, with relaxed constraints, most of the arcs are feasible, so the algorithm cannot significantly reduce the search area. However, with relaxed constraints the solution becomes easier and even a naive heuristic can find a good solution. Figure 8 represents the rate of optimal solution that

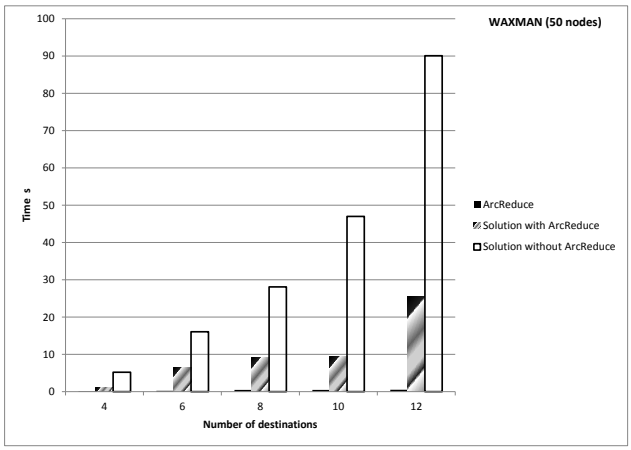

Figure 7: Execution time with/without using ArcReduce.

can be found by an algorithm only designed to find the optimal Steiner tree (we have used an existing ILP formulation of Steiner tree) [15] without considering the QoS constraints.

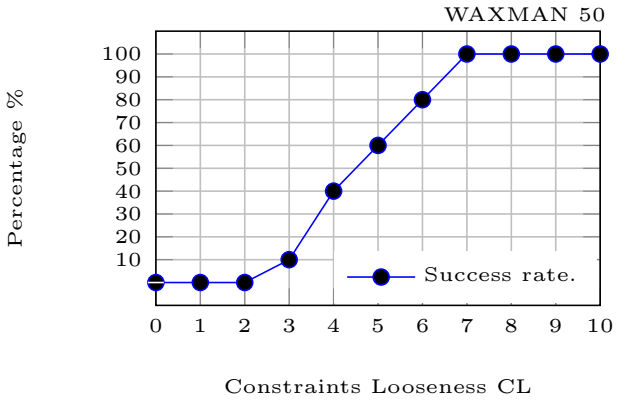

Figure 8: The success rate using an exact Steiner algorithm.

7.4. Connectivity

We highlight in this part the connectivity of the graph obtained after the filtering step, especially the connectivity of the multicast members. For this purpose we extend the ArcReduce algorithm to check whether there is at least a path from the source to each destination using breadth-first search strategy. The connectivity test allows us to know if a given problem has a solution or not before searching for the solution.

With simple modification of this connectivity test, we can save the set or the partial set of destinations which are reachable from the source, to decide later if we want to solve the problem partially or not.

8. Conclusion and Future Work

In this paper, we solved the multicast routing subject to multiple constraints with minimal cost, which is an NP-hard problem. The optimal solution, when it exists, is not always a tree but a more complex structure called hierarchy. We have also proposed a new pretreatment algorithm which is efficient to reduce the search space. The Integer Linear Program solving this problem is not trivial, because the solution is different from a sub-graph. The proposed ILP finds the optimal solution when it exists. This allows us to evaluate the proportion of instances in which the solution is not a tree (which is often considered to be the only possible solution). Even if the computation time is reasonable in real instances and independent of the number of constraints, it is of course not tractable when the size of the instances or the number of destinations grows. ArcReduce pretreatment has been proposed to accelerate and increase the resolution capacity of our ILP in large datasets. Our results show that the proposed pretreatment is efficient in hard computational cases, when the QoS requirements are strict. Notice that the pretreatment can also be applied with success for heuristics.

One of the main interest of our exact solution is that it allows us to know the real performances of the heuristics proposed to solve this problem. In the examined benchmark networks, the evaluation tests using the well known heuristic MAMCRA show that the solution returned by the heuristic is at most 20.6% larger than the optimal solution in the worst case (and most often around 8.5% larger).

For real applications, this work may be pursued further. A first idea would be to improve the first ILP formulation. Proposing a fast heuristic which takes into account all what has been concluded in the experiments part may also be challenging. Since the proportion of instances whose optimal solution is a hierarchy and not a tree seems to be strongly related to the topology of the network (and especially to the connectivity), it would be interesting to know which parameters are the most influent.

References

[1] S. Ehsan and B. Hamdaoui. A Survey on Energy-Efficient Routing Techniques with QoS Assurances for Wireless Multimedia Sensor Networks. *IEEE Communications Surveys Tutorials* 265-278, 2012.

[2] X. Masip-Bruin and M. Yannuzzi and J. Domingo-Pascual and A. Fonte and M. Curado and E. Monteiro and F. Kuipers and P. Van Mieghem and S. Avallone and G. Ventre and P. Aranda-Gutirrez and M. Hollick and R. Steinmetz and L. Iannone and K. Salamatian. Research challenges in QoS routing. *Computer Communications* 563-581, 2006.

[3] R. Karp. Reducibility among combinatorial problems. In RaymondE. Miller, JamesW. Thatcher, and JeanD. Bohlinger, editors, *Complexity of Computer Computations*, The IBM Research Symposia Series, pages 85-103. Springer US, 1972.

[4] H. Takahashi and M. Matsuyama. An approximate solution for the Steiner problem in graph. *Math. Japonica* 24(6): 573-577, 1979.

[5] L. Guo and I. Matta. QDMR: an efficient QoS dependent multicast routing algorithm. *Real-Time Technology and Applications Symposium, 1999. Proceedings of the Fifth IEEE*, pages 213-222, 1999.

[6] K. Tsai and C. Chen. Two algorithms for multi-constrained optimal multicast routing. *International Journal of Communication Systems*, 16(10):951-973, 2003.

[7] Zheng Wang and J. Crowcroft. Quality-of-service routing for supporting multimedia applications. *IEEE Journal on Selected Areas in Communications*, 7(14):1228-1234, 1996.

[8] F. Kuipers and P. Van Mieghem. MAMCRA: a constrained-based multicast routing algorithm. *Computer Communications*, 25(8):802-811, 2002.

[9] H. Van Mieghem, P. De Neve and F. Kuipers. Hop-by-hop quality of service routing. *Computer Networks*, 37(34):407-423, 2001.

[10] M. Sun and Q. Zhang and W. Cao. An Improved QoS Multicast Routing Algorithm Based on Chaotic Neural Network. *International Conference on Intelligent Human-Machine Systems and Cybernetics* 139-143, 2015.

[11] Mohammad Ali Raayatpanah. Multicast routing based on data envelopment analysis with multiple Quality of Service parameters. *International Journal of Communication Systems* 139-143, 2015.

[12] M. Molnar, A. Bellabas and S. Lahoud. The cost optimal solution of the multi-constrained multicast routing problem. *Computer Networks*, 56(13):3136-3149, 2012.

[13] B. Waxman. Routing of multipoint connections. *Selected Areas in Communications, IEEE Journal on*, 6(9):1617-1622, 1988.

[14] B. Baran and R. Sosa. A new approach for AntNet routing. *Computer Communications and Networks* 303-308, 2000.

[15] Michel X. Goemans, Young-Soo Myung. A catalog of steiner tree formulations. *Networks* 19-28, 1993.

[16] Hwang, Frank K and Richards, Dana S and Winter, Pawel and Widmayer, P. The Steiner Tree Problem. *Annals of Discrete Mathematics* 382-383, 1995.

[17] Wang, Bin and Hou, Jennifer C. Multicast routing and its QoS extension: problems, algorithms, and protocols. *IEEE network* 22-36, 2000.

[18] Diego Kreutz and Fernando M. V. Ramos and Paulo Veríssimo and Christian Esteve Rothenberg and Siamak Azodolmolky and Steve Uhlig Software-Defined Networking: A Comprehensive Survey. *Proceedings of the IEEE* 14-76, 2015.

[19] A.T. Haghighat and K. Faez and M. Dehghan and A. Mowlaei and Y. Ghahremani. GA-Based Heuristic Algorithms for QoS Based Multicast Routing. *Knowledge-Based Systems.* 305-312, 2003.

[20] Baolin Sun and Shangchao Pi and Chao Gui and Yue Zeng and Bing Yan and Wenxiang Wang and Qianqing Qin. Multiple constraints QoS multicast routing optimization algorithm in MANET based on GA. *Progress in Natural Science.* 331-336, 2008.

[21] Ya-li WANG and Mei SONG and Yi-fei WEI and Ying-he WANG and Xiao-jun WANG. Improved ant colony-based multi-constrained QoS energy-saving routing and throughput optimization in wireless Ad-hoc networks. *The Journal of China Universities of Posts and Telecommunications.* 43-59, 2014.

[22] Hua Wang and Hong Xu and Shanwen Yi and Zhao Shi. A tree-growth based ant colony algorithm for QoS multicast routing problem. *Expert Systems with Applications.* 11787-11795, 2011.

[23] N.B.Ali, M.Molnar, A.Belghith. Multiconstrained QoS Multicast Routing Optimization. *Technical Report 1882, INRIA.* 2008.

[24] Wei Li, Kangshun Li, Ying Huang . A EA- and ACA-based QoS multicast routing algorithm with multiple constraints for ad hoc networks. *Soft Computing.* 57175727, 2017.

[25] Ajay Kumar Yadav, Santosh Kumar Das, Sachin Tripathi. EFMMRP: Design of efficient fuzzy based multi-constraint multicast routing protocol for wireless ad-hoc network. *Computer Networks.* 15-23, 2017.

[26] T. Lu and J. Zhu. Genetic Algorithm for Energy-Efficient QoS Multicast Routing. *IEEE Communications Letters.* 31-34, 2013.

[27] Naik A., Parvathi K., Satapathy S.C., Nayak R., Panda B.S. QoS Multicast Routing Using Teaching Learning Based Optimization. *Advances in Intelligent Systems and Computing.* 2013.

[28] W. A. Sheikh and A. R. Bashandy and A. Ghafoor. A combined approach for QoS based multicast routing and resource allocation. *IEEE International Conference on Communications, 2005.* 136-142, 2005.