QoS multicast aggregation under multiple additive constraints

N. Ben Ali a,*, A. Belghith a, J. Moulieca, M. Molnarc

aHANA Research Group, Ecole Nationale des Sciences de l’Informatique, University of Manouba, Tunisia
bIRISA, University of Rennes 1, France
cIRISA, INSA, France

Article Info
Article history:
Received 27 September 2007
Received in revised form 27 May 2008
Accepted 27 May 2008
Available online 17 June 2008

Keywords:
Multicast
Tree aggregation
QoS routing
Additive constraints
Cycle reduction

ABSTRACT
IP Multicast has been proposed in order to manage group communications over the Internet in a bandwidth efficient manner. Although such a proposition has been well studied, there are still some inherent problems for its widespread deployment. In this paper, we propose a new algorithm coined mQMA that deals with the two main problems of traditional IP multicast, i.e., multicast forwarding state scalability and multi-constrained QoS routing. The algorithm mQMA is a QoS multicast aggregation algorithm which handles multiple additive QoS constraints. It builds few trees and maintains few forwarding states for the groups thanks to the technique of multicast tree aggregation, which allows several groups to share the same delivery tree. Moreover, the algorithm mQMA builds trees satisfying multiple additive QoS constraints. We show via extensive simulations that mQMA reduces dramatically the number of trees to be maintained and reduces the utilization of the network resources, yet it leverages the same overall QoS performances as Mamcra which is the main known multi-constrained multicast routing algorithm.

© 2008 Elsevier B.V. All rights reserved.

1. Introduction
Recently, the Internet has shown a tremendous growth. Emergent multimedia applications like audio/video conferencing, video on demand, IP-telephony usually have other requirements than traditional Internet services such as e-mail and file transfer services. Moreover, these applications may involve a set of users and not only a sender and a receiver.

IP multicasting techniques have been proposed to support group communications over the Internet with the aim of reducing the network resource consumption. Although IP multicast [1] has been studied for a long time, it is not totally well deployed over the Internet. The multicast forwarding state scalability problem and the lack of QoS support can be considered among reasons for IP multicast not to be widely deployed. In traditional IP multicast, data for a given group is forwarded through a tree structure covering the members of this group. The on-tree routers must maintain a per-group forwarding state. With the recent evolution of Internet applications, the number of groups may be very large which, in turn, increases considerably the number of forwarding states. Large forwarding tables induce large memory requirements and slow down the address look-up process. In unicast, the aggregation of IP addresses has been possible since they are considered and hence treated in a hierarchical manner. In IP multicast, such an aggregation is much more difficult as multicast addresses cannot convey any location information. Moreover, maintaining multicast trees requires an exchange of control messages causing an important overhead as the number of multicast groups grows. This represents the problem of forwarding state scalability [2,3]. This problem should be solved before a large number of groups can be accommodated.

Nowadays, Internet applications require some QoS awareness. The best-effort functioning provided by IP-based network does not satisfy application needs. In addition to traffic engineering mechanisms, QoS oriented routing algorithms are needed to meet the application QoS requirements.

Multicast tree aggregation [4–6] is a recent approach that deals with the problem of multicast forwarding state scalability. In this approach, multiple groups share the same delivery tree within a domain whereas in traditional IP multicast, a tree is built and maintained for each group. With multicast tree aggregation, fewer trees are maintained which amounts to reduce both the number of forwarding states in routers and the overhead induced by control messages. Some information has to be added in the border routers of the domain in order to multiplex the data for the groups onto the aggregated tree. Several algorithms have been proposed to perform tree aggregation: AM (Aggregated Multicast, [4]) and STA (Scalable Tree Aggregation,
Proposed algorithm mQMA in three main steps. Section 4 shows that TALD (Tree Aggregation in Large Domains, [10]) achieve good aggregation ratio in large domains where traditional multicast aggregation algorithms fail.

However, these proposed algorithms aggregate multicast groups without considering any QoS requirements. Aggregation is based on trees computed by IP multicast routing protocols which use the shortest path tree algorithm optimized on one single metric, typically the hop count. Two solutions have been proposed in the literature to deal with QoS multicast aggregation: AQoS (Aggregated QoS Multicast, [11]) and Q-STA (QoS Multicast Tree Aggregation, [12]). The goal of these two algorithms is to aggregate groups to trees while respecting bandwidth requirements of the different groups. However, multicast applications today need to satisfy more than one or two QoS criteria and consequently multiple QoS multicast aggregation is needed. Tree shared by multicast members must fulfill flow requirements such as delay, bandwidth, delay variation (also known as jitter), packet loss and/or number of hops. To achieve QoS multicast aggregation under several QoS constraints, the forwarding structure must be computed by a multi-constrained multicast routing algorithm. In most cases, this structure is not a tree for it contains some cycles. Most proposed algorithms perform single or dual QoS multicast routing [13]. For the multi-constrained multicast routing problem which involves multiple QoS metrics, few algorithms have been proposed.

In this paper, we propose a new algorithm called mQMA (multi-constrained QoS Multicast Aggregation) which performs multicast aggregation taking into account multiple QoS requirements. To the best of our knowledge, it is the first proposed protocol that deals with these two main issues of multicast forwarding scalability and of multi-constrained multicast routing. Our protocol mQMA is based on two principles. First, the technique of multicast tree aggregation reduces the number of trees to be maintained and second mQMA builds trees satisfying several additive constraints.

The paper is organized as follows: Section 2 presents previous work that dealt with these two fields. Section 3 details our proposed algorithm mQMA in three main steps. Section 4 shows how mQMA reduces the forwarding structure by eliminating as many redundancies as possible. Section 5 presents and analyzes the simulation results. Finally, we conclude our work in Section 6.

2. Related work

This section describes the related work concerning the two important domains related to multicast; namely multicast aggregation and multi-constrained multicast QoS routing.

2.1. Multicast tree aggregation

2.1.1. Multicast tree aggregation principles

Multicast tree aggregation is proposed as a solution to the multicast forwarding state scalability problem. It is a multicast scheme which forces multicast groups to share the same delivery tree within a domain. Data packets for different groups are delivered via the same distribution tree called the aggregated tree. This induces less tree construction and hence less forwarding states to be stored in the on-tree routers. This way, routers within a domain need to keep just one state per aggregated tree and not per multicast group. This protocol can replace an intra-domain multicast routing protocol.

Routing messages on an aggregated tree can be no longer performed using the original group multicast IP address. Each aggregated tree is assigned a label and the edge routers maintain the group-label matching in a group-label table. This group-label matching can be achieved either through MPLS labels or through IP encapsulation. The label represents either a MPLS label or a multicast address of a group which is not active in the domain: a virtual multicast address if the IP encapsulation technique is adopted. Indeed, if multicast aggregation is deployed in a MPLS domain, MPLS labels are used and distributed with LDP [14]. With IP encapsulation, the packets are forwarded with a multicast routing protocol according to the virtual multicast address added to the packet.

An edge router, at the domain entry, pushes a label into incoming multicast packets and keeps a group-label table. Within the domain, core routers use this label to route packets and only maintain forwarding states for these labels. When leaving the domain, the edge router removes the label and the packet is forwarded in the usual way. This label-based mechanism is depicted in Fig. 1 which represents a domain composed of four edge routers $b_1, b_2, b_3$ and $b_4$. We consider here three different groups $g_1, g_2$ and $g_3$: each having members attached to edge routers $(b_1, b_2$ and $b_4)$. Without aggregation, three trees are built and maintained in the domain as shown in the top part of the figure. However, when tree aggregation is deployed, only one tree, say $t_1$, is needed and then built for these three groups as shown in the bottom part of Fig. 1. This tree is assigned label $l_1$ and all three groups are mapped to this label in the group-label table of $b_1$.

2.1.1.1. Group-tree matching

In aggregated multicast, groups must be matched to aggregated trees. The group-tree matching can be done in various ways as defined in [15]. The matching for a given group is said to be a perfect match if the tree covers exactly the group members and the tree cost (in terms of edge number) is no more than the cost of the native tree. It may also be a leaky match, if there are tree leaves (routers) that do not actually serve members of this group. In this case, additional bandwidth is used to deliver data to nodes that are not in this multicast group. However, this bandwidth waste is usually not high. The group-tree matching is said to be an incomplete match if the tree does not cover all the members of this group, that is some members are left uncovered. Finally it may be an incomplete leaky match if there are tree leaves that do not serve members and yet some members are not covered by the aggregated tree. All these aggregation types are illustrated in Fig. 2. When aggregating $g_1$, tree $t$ achieves a perfect match as it covers exactly all the members. Multicast group $g_2$ has only $b_1$ and $b_2$ as multicast members, so, when $t$ is used to aggregate $g_2$, the aggregation is leaky as the router $b_4$ will receive packets unnecessarily. If $t$ is used to aggregate multicast group $g_3$, having $b_1, b_2, b_3$ and $b_4$ as multicast members, the aggregation is incomplete. To forward data to member $b_4$, a tunneling-based mechanism could be used. When the multicast group $g_4$ is considered, the aggregation with tree $t$ is an incomplete leaky match, a tunnel could be used to forward data to $b_3$, yet $b_4$ receives multicast data unnecessarily.

The choice of the aggregation type depends on what will be privileged bandwidth utilization or number of entries in routers. Perfect match aggregation reduces the number of states without any overhead in terms of additional bandwidth. Adopting a leaky match aggregation with a bandwidth threshold reduces further the number of trees. In the extreme case, if the bandwidth threshold is set to a high value, only one tree is needed for all the multicast groups but too much bandwidth is wasted.
2. Multicast tree aggregation proposals

Several protocols have been proposed dealing with multicast tree aggregation.

2.1. Centralized multicast aggregation. The first protocol is AM and is proposed in [4]. In AM, a centralized entity called tree manager performs the group-tree matching by assigning labels to groups. This tree manager stores the information concerning the group memberships and the trees maintained in the domain. Whenever a multicast group arrives, the edge router contacts the centralized entity which is in charge of finding an adequate tree for the given group. In this phase, if the tree manager finds a tree respecting the bandwidth assumed threshold, then the aggregation is performed. If not, the tree manager configures a new tree for the group. This new tree will be a candidate for further aggregations. In all the cases, the tree manager informs the border router of the label found. Then, the edge router is able to route packets for the group onto the aggregated tree. The protocol STA is proposed in [5] to speed up the aggregation algorithm performed by the tree manager. The ideas developed are a fast selection function based on an efficient sorting of the trees and a low storage of the trees.

2.2. Distributed multicast tree aggregation. BEAM [7] and DMTA [8] are proposed to overcome the drawbacks of centralized aggregation. In BEAM, the task of the tree manager is distributed among several routers, called core routers. Each core router has a local view of the configured tree in the domain. This protocol re-
2.2. Multi-constrained multicast routing

The problem of QoS routing, even in the unicast case, is known to be NP-complete and has been extensively studied by the research community [13]. Before presenting the mechanisms used to construct multi-constrained QoS multicast delivery structures, we need first to specify some hypotheses used to solve these problems and the notation used throughout this paper. Then we present existing approaches to achieve multicast QoS routing.

2.2.1. Hypotheses

QoS routing approaches assume that the network-state information is temporarily static and has been distributed throughout the network. This network-state information is accurately maintained at each node and we assume that this information can be collected by any appropriate traffic engineering mechanism. The QoS metrics are categorized into additive (e.g., delay, jitter, etc.) and min (resp. max) metrics (e.g., bandwidth). The constraints on min (resp. max) QoS measures can be easily treated by pruning all links (and disconnected nodes) which do not satisfy the requested min (resp. max) QoS constraints. In contrast, constraints on additive QoS measures cause more difficulties. Hence, without loss of generality, all QoS measures are assumed to be additive.

A network topology is modeled as an undirected graph $G = (V,E)$, where $V$ is the set of vertices representing the network nodes and $E$ is the set of edges representing the network links. Each link is characterized by $m$ additive QoS metrics. So, we associate to each link an $m$-dimensional link weight vector of $m$ non-negative QoS weights ($w^i_j$ for $i = 1, 2, \ldots, m$ and $j$ is a link in $E$). The $m$ QoS constraints (limits) $l^i_j$ for $i = 1, 2, \ldots, m$ are represented by the constraint vector $L$. A multicast group $g$ is composed of a source $s$ and a set of members $D = \{d_1, d_2, \ldots, d_k\}$, where $k$ is the number of multicast members.

2.2.2. Multi-constrained multicast routing proposals

For the multicast case, a number of QoS routing algorithms based on single, dual and multiple metrics have been proposed. Single metric QoS multicast routing algorithms have been proposed for cost [16,17] and delay [18,19]. Dual metric-based routing algorithms have been proposed for the following combinations: cost-delay [20,21] and delay-jitter [22,17]. Few works, however, have dealt with multiple QoS metrics. One way to tackle the multicast QoS routing problem is to compute a set of unicast paths from the source to each of the multicast members using a unicast QoS routing algorithm. The set of these computed paths form a sub-graph that should be reduced to optimize network utilization without violating the constraints. This strategy is adopted by the Multicast Adaptive Multiple Constraints Routing Algorithm (Mamcra) which is proposed by [23] as the multicast extension of Sambra, a unicast QoS routing algorithm [24] and the Taboo-based QoS Multicast Routing Algorithm (Taboo-QMR) [25].

Mamcra proceeds in two phases: path computation and path reduction. In the first phase, Mamcra computes multi-constrained shortest paths from the source node to each destination using Sambra, the QoS unicast routing algorithm (detailed in Appendix A). In the second phase, the sub-graph representing the set of obtained paths is then optimized by eliminating as many cycles as possible without violating the different constraints. Mamcra proposes a heuristic approach based on a greedy algorithm to solve this problem which is efficient but the quality of the approximation is not proved.

In [25], the authors propose Taboo-QMR, a QoS multicast routing algorithm where they adopt the same strategy to achieve the first phase using any unicast QoS routing algorithm. They propose improvements to the reduction phase of Mamcra by adopting a meta-heuristic approach based on taboo search algorithm to provide a sub-optimal solution. Simulations performed on large networks (300–650 nodes) showed that Taboo-QMR eliminates more cycles than Mamcra and finds better solutions than Mamcra according to the same objective function used by both algorithms.

3. The mQMA algorithm

In this section, we describe our proposed mQMA algorithm to achieve multi-constrained multicast routing. In addition to the
hypotheses and notations specified in the previous section, we need the following definitions.

3.1. Definitions

Given a graph \( G = (V,E) \), where \( V \) is a set of vertices and \( E \) a set of edges, the path \( p(s,d) = (P_c, P_d) \), where \( P_c \subseteq V \) and \( P_d \subseteq E \) are the sets of vertices and edges of the \( G \) connecting node \( s \) to node \( d \). Consider a tree, \( T = (T_v, T_e) \), where \( T_v \) is the set of vertices included in \( V \) and \( T_e \) is a set of edges included in \( V \). Consider a multicast group \( g \), as a set of nodes of \( G \), that is \( g \subseteq V \).

**Definition 1.** A path \( P = (P_c, P_d) \) can be crafted on a tree \( T \), if the obtained graph \( U = (U_v, U_e) \) is a tree where \( U_v = P_c \cup T_v \) and \( U_e = P_d \cup T_e \). Let \( GFPS \) denotes the grafting operator, then the grafting of \( P \) to \( T \) is denoted: \( U = P \cup GFPS \).

**Definition 2.** The set \( S = g \cap T \) denotes the nodes that are both in group \( g \) and in tree \( T \).

**Definition 3.** Given a multicast group \( g \) and a set of \( m \) additive constraints \( L = (l_1, l_2, \ldots, l_m) \), the Feasible Paths Set, \( FPS \), refers to the set of paths from the source node to the multicast destination nodes that fulfill all the defined constraints.

3.2. Algorithm mechanisms

The algorithm mQMA achieves multicast tree aggregation in a centralized way: a tree manager maintains a Multicast Tree Set (MTS) which contains all the multicast trees configured so far into the domain. We consider here a centralized aggregation and propose an algorithm to perform aggregation under multiple additive constraints. A centralized tree manager, however, creates well-known bottleneck problems. Enhanced versions of mQMA may adopt distributed aggregations in the way of DMTA. Furthermore, TALD techniques could be adapted and integrated to mQMA when deployed in large domains. All these issues require indeed further investigations, however, we consider them beyond the scope of this paper as we need first to exacerbate hypotheses and notations specified in the previous section, we need the following definitions. In the following, we detail each of the three steps of leaky match aggregation, the bandwidth loss threshold must be specified. In the following, we detail each of the three steps of the algorithm mQMA.

3.2.1. Step 1. Path computation

The aim of this step is to compute multi-constrained paths from the source node to each destination node of the group. To perform this step, any unicast QoS routing algorithm can be used. For the simulations and in order to compare mQMA to Mamcra, the algorithm Samcra [24] is used. This first step is mandatory to determine the multi-constrained FPS (FPS) for the current group. The quality of the paths in FPS depends, however, on the unicast routing algorithm used and is induced by its objective function. If Samcra is used, feasible paths from source node to each destination node are found if they exist and obey to its objective function. A group member for which no path is found is removed from the group.

3.2.2. Step 2. Tree decomposition

Let \( GFPS \) be the graph obtained from the collection of the paths in the computed FPS for the current group. Recall that \( GFPS \) is not necessarily a tree. The aim of this step is to decompose \( GFPS \) into several trees. The obtained trees will form the FTS. If the \( GFPS \) contains no cycle, the FTS contains only one tree: \( FTS = GFPS \) and the algorithm goes directly to step 3. In the worst case, the FTS contains \( |FPS| \) trees, where \( |FPS| \) denotes the number of paths in \( FPS \). If the \( GFPS \) contains cycles, the tree decomposition tries progressively in a greedy manner to graft the paths in \( FPS \) onto trees currently in \( FTS \). A path is added to the \( FTS \) as a new tree only if its grafting creates cycles with all trees within the current \( FTS \). The adopted greedy approach goes as follows. First, we order the paths in \( FPS \) according to the number of their members in non-increasing order. Paths having the same number of members, are sorted in non-increasing order of their quality as provided by the objective function of the used unicast algorithm. We note here that any sorting criterion may be well used. The choice of this criterion and its implications are beyond the scope of this paper and will be investigated in further work. The \( FTS \) is initialized by the first path of the sorted \( FPS \) that is the one covering the largest number of members. This path forms the first tree added to the (currently empty) \( FTS \). The remaining paths are considered in the prescribed order for grafting to current trees in \( FTS \). A path is only considered for grafting if and only if it contains members not already included in the current trees of the \( FTS \). If all the members served by this path are part of the current trees of the \( FTS \), this path is not considered for grafting. In fact, feasible paths (but not necessarily optimal) for the members of the current considered path could be found in the current trees of the \( FTS \). We may here also question as to how to order trees in the \( FTS \). In this paper, these trees are considered according to the order of their insertion in the \( FTS \). Once again, it is beyond the scope of this paper to investigate the performance of different choices.

3.2.3. Step 3. Aggregation

After executing the above two steps, the tree manager attempts to achieve a global aggregation of the whole multicast group. This consists to find from the \( MTS \) a multicast tree, say \( MT_i \), that includes all the members covered by the \( FPS \) without violating the constraints. For the case of a perfect match aggregation, the aggregated tree \( MT_i \) must have a cost (in terms of the number of links) at most the cost of all the trees (if several) in the \( FTS \). For the case of a leaky match aggregation, the cost must not exceed the bandwidth threshold (expressed as a percentage). Note here that, when a perfect match is adopted, the aggregated tree may have a lower cost than that of the \( GFPS \) structure. Recall that the multi-constrained multicast routing algorithm optimizes an objective function expressed in terms of the multi-constrained metrics (the Samcra’s objective function when Samcra is used) rather than the number of links. If \( MT_i \) does exist, it is used to aggregate the multicast group \( g \). However, if \( MT_i \) is not found, the aggregation depends on whether the \( FTS \) contains one or several trees. If \( FTS \) contains just one tree, this tree is then added to the multicast tree set \( MTS \). However, if the \( FTS \) contains more than one tree, mQMA proceeds by a partial aggregation. Recall that each tree \( FT_i \) \((1 \leq i \leq |FTS|)\) of the \( FTS \) covers a sub-group \( g_i \) of the multicast group \( g \). The partial aggregation tries to aggregate independently each sub-group \( g_i \). When no aggregation is possible for a given sub-group, its corresponding tree is added to the \( MTS \). Note here that the order in which we consider the sub-groups has no effect on the aggregation. Partial aggregation results in a set of aggregated trees. Subsequently, the tree manager updates the routing information and informs all the involved routers.

Algorithm 1 describes formally mQMA during the two first steps while Algorithm 2 describes the third step of mQMA.
Algorithm 1. The multi-constrained QoS Multicast Aggregation protocol (mQMA): step 1 and step 2

**Input:** The network $G = (V, E)$, a group $g = (d_1, d_2, \ldots, d_{|g|})$ having source $s$ and the constraint vector $\bar{L}$.

**Output:** The set $FTS$ covering, if possible, all members of $g$.

**Step (1): Path Computation**

$FPS \leftarrow \emptyset$;

For $i$ from 1 to $|g|$ do

- Compute an optimal path $P_i$ from source $s$ to destination member $d_i$ using a given unicast QoS routing algorithm;
- Add $P_i$ in $FPS$;

end For

**Step (2): Tree Decomposition**

$FTS \leftarrow \emptyset$;

If ($G_{FPS}$ contains no cycles) then

$FTS = G_{FPS}$;

else

- Arrange the $FPS$ in non increasing order of the number of members covered by each path. Paths having the same number of members, are sorted in non increasing order of their quality as provided by the objective function of the unicast algorithm.

For $i$ from 1 to $|FPS|$ do

- If ($P_i$ contains members not yet in the $FTS$) then

  - $Grafted = false; j = 1$;
  
  - While ($Grafted$ is false and ($j \leq |FTS|$)) do

    - If ($FT_j \cup g P_i$ doesn’t form a cycle) then

      - $FT_j = FT_j \cup g P_i$;
      
      - $Grafted = true$;

    end If

  end While

- $j += 1$;

- If ($Grafted$ is false) then

  - Add $P_i$ to $FTS$ as $FT_{|FTS|+1}$;

end If

end For

end If
Algorithm 2. Step 3: Aggregation

**Input:** The network $G = (V, E)$, a group $g$ with source $s$, the constraint vector $\mathbf{L}$, the bandwidth threshold $t_b$, the current multicast tree set $MTS$, the $FTS$ computed in the step 2.

**Output:** The updated multicast tree set $MTS$.

**Step (3): Aggregation**

$\text{candidates} \leftarrow \emptyset$;

For $i$ from 1 to $|MTS|$ do

If $(MT_i$ can cover $g$ and fulfills the constraints and $\sum_{1 \leq k \leq |FTS|} \text{cost}(MT_i) - \sum_{1 \leq k \leq |FTS|} \text{cost}(FT_k) \times t_b)$ then

Add $MT_i$ to $\text{candidates}$;

end If

end For

If ($\text{candidates} \neq \emptyset$) then

Aggregate $g$ to the tree $t$ among the $\text{candidates}$ of minimum cost;

else

If ($|FTS| = 1$) then

Add $FT_1$ to the $MTS$;

else

For $i$ from 1 to $|FTS|$ do

Let $g_i = FT_i \cap g$ the subset members covered by tree $FT_i$;

For $j$ from 1 to $|MTS|$ do

If $(MT_j$ can cover $g_i$ and fulfills the constraints and $\text{cost}(MT_j) - \text{cost}(FT_i) \leq \text{cost}(FT_i) \times t_b)$ then

Add $MT_j$ to $\text{candidates}$;

end If

end For

If ($\text{candidates} \neq \emptyset$) then

Aggregate $g_i$ to the tree $t$ among the $\text{candidates}$ of minimum cost;

else

Add $FT_i$ in $MTS$

end If

end For

end If

end If
3.3. mQMA on an example

Let us consider the topology presented in Fig. 3 composed of seven nodes and seven links where s is the source node and nodes d₁ and d₂ are members of the multicast group. Two additive metrics \((w₁, w₂)\) are considered and the constraint limits are fixed to \((L₁, L₂) = (7, 7)\).

3.3.1. Step 1. Path computation

The execution of the first step of mQMA provides the set \(FPS = \{(s-b-c-e-d₁), (s-a-c-e-d₂)\}\) of shortest paths to d₁ and d₂. These paths are computed using Samcra algorithm; recall that any other unicast QoS routing can be used.

3.3.2. Step 2. Tree decomposition

During the second step of mQMA, the structure \(G_{FPS}\) contains one cycle. The tree decomposition step gives the set \(FTS = \{FT₁ = (s-b-c-e-d₁), FT₂ = (s-a-c-e-d₂)\}\).

3.3.3. Step 3. Aggregation

The last step of mQMA aims to find trees in the multicast tree set \(MTS\) covering members of the current group and fulfilling the constraints. For our example, the current content of the MTS induces whether or not global aggregation is possible.

- If the MTS contains for example a tree covering both d₁ and d₂ and fulfilling the constraint limits, global aggregation is possible. For example, the MTS contains the tree \(MT₁ = \{(s-b-c-e-d₁), (s-b-c-e-d₂)\}\). This tree is then used to aggregate the group \((d₁, d₂)\).

- If global aggregation is not possible, we proceed to a partial aggregation. This is exactly the case where the MTS contains no one tree which covers both d₁ and d₂.

4. mQMA: a proposal to enhance multi-constrained routing structure

The mQMA algorithm performs multicast aggregation taking into account multiple members requirements. mQMA builds trees satisfying multiple constraints and reduces efficiently the forwarding structure. In this section, we first detail the cycle problem and the way it is handled by existing proposals. Then, we clarify how mQMA contributes in the reduction of forwarding structures.

4.1. Problem statement

Multicast routing aims to forward multicast data efficiently by sending single packets through the shared links and duplicating them if it is necessary through a tree structure. When multi-constrained multicast routing is considered, this philosophy is no longer respected. Indeed, multi-constrained multicast routing computes a routing structure that satisfies multiple additive QoS metrics and this structure is not compulsorily a tree. Not to affect the multicast philosophy, multi-constrained proposals take into account the reduction of the computed routing structure in order to eliminate as many redundancies as possible without violating the members requirements. So, in addition to computing feasible paths to multicast destinations, a good multicast proposal should try to reduce the obtained routing structure to leverage the traditional tree structure by eliminating as more cycles as possible. In the rest of this paper, we refer to this problem as the cycle problem.

4.1.1. Cycle problem on an example

Fig. 3 illustrates the cycle problem. If the constraint limits are \((7, 7)\), optimal paths \(FP₁\) and \(FP₂\), respectively, serving destinations d₁ and d₂, form the cycle \((s-a-c-b-s)\). Observe here that \((s-b-c-e-d₂)\) is a feasible path for node d₂ and can replace \(FP₂\) which then amounts the eliminating the cycle. In the same manner, the path \((s-a-c-e-d₁)\) is a feasible path for d₁ and can replace \(FP₁\) which then amounts also to remove the cycle. However, when the constraint limits are fixed to \((6, 6)\), the path \((s-a-c-e-d₁)\) does not respect the constraints for d₁ and the path \((s-b-c-e-d₂)\) does not respect the constraints for d₂. Consequently, the cycle \((s-a-c-b-s)\) cannot be removed. In this case, c receives data packets twice, once from a and once from b and the link \((c-e)\) transfers duplicated packets.

A specific routing has to be in place as traditional IP routing cannot support that case.

4.2. Existing proposals to solve cycle problem

Mamcra algorithm as presented in Section 2.2.2 and detailed in Appendix A, proceeds in two steps: the first step consists on computing individual paths from source node to destination nodes and the second step is a reduction procedure (step B detailed in Algorithm 3) that aims to eliminate as many cycles as possible from the routing structure found in the first step. Mamcra considers the computed paths in non-increasing order of the number of members. During the reduction step, Mamcra builds the routing structure progressively by adding the optimal paths one at a time. If the added path forms a cycle with the current routing structure, Mamcra tries to perform a reduction if possible otherwise this cycle persists. Authors in [25] shows that this reduction procedure presents some drawbacks that are essentially due to the order adopted when removing cycles. In fact, the order adopted in the greedy procedure influences elimination of cycles. The authors in [25] proposed improvements to the reduction phase of Mamcra by adopting a meta-heuristic approach based on a taboo search algorithm to provide a sub-optimal solution. Using a taboo-based approach is interesting whenever the graph topology is large and complex.

![Fig. 3. Computing the FPS using the constraints (7,7), for group \((d₁, d₂)\) having source s.](image-url)
the number of cycles is important. When dealing with small topologies, taboo search-based approach is rather expensive to use.

4.3. mQMA contribution to solve cycle problem

The mQMA algorithm reduces the routing entries but also can be considered as an alternative to solve the cycle problem. In addition to the aggregation benefit, the routing structure used by mQMA to deliver multi-constrained multicast data has been implicitly reduced. The algorithm mQMA removes cycles during its two last steps:

1. **During the tree decomposition:** when a path is considered for grafting into an existing FTS, it can cover some members not already considered. That means that this path is optimal for one destination but may be feasible and not optimal for other destination nodes. In that case, optimal paths for these multicast destination nodes which are not yet processed by the tree decomposition won’t be considered for grafting. In fact, as specified in Section 3.2, a path of the FPS is considered for grafting only if it contains destination nodes that are not in the trees of the FTS. If the optimal paths for these members were forming some cycles, then with the strategy adopted to achieve tree decomposition, these cycles are readily removed.

2. **During the aggregation step:** one global tree that is feasible and that covers the members can exist even if the structure FTS contains more than one tree. In that case, the cycles are removed from the group quite easily as global aggregation is made.

4.3.1. Elimination of the cycles by mQMA on an example

To apprehend how mQMA eliminates cycles during the two steps as described above, let us consider in the following two topologies examples.

4.3.1.1. Cycles reduction during the tree decomposition step. To show how cycles are removed during this step, we will consider the example on Fig. 4. Let us consider the example presented in Fig. 6 with the source node and nodes d0, d1, d2, the multicast members. If the constraints are fixed to (6,6), and if SAMcra is used as the unicast QoS routing, the first step of mQMA algorithm returns the set FPS1 = \{ (s–d0), (s–a–d0–d1) \}. The FTS contains the cycle (s,a,c,b,s). mQMA second step starts by considering the path covering the most number of members. The two paths have the same number of members and have the same objective function value. Assume that the path for d1 is considered and added to the FTS as a first tree FT1 = { s–b–c–d1 }. Then, the path for d2 is considered for grafting, it contains d2 which is not yet in the FTS but this path can not be grafted to FT1, so, it is added to the FTS as a second tree FT2 = { s–a–c–d2 }. At the end of this second step, the FTS contains two trees FT1 and FT2. The last step of mQMA attempts firstly to perform global aggregation. If the FTS contains either the path MT1 = { (s–a–c–d1), (s–a–c–d2) } or the path MT2 = { (s–a–b–d1), (s–a–b–d2) }, then this tree can be used to aggregate the group as it covers the members, respects the members requirements and does not exceed the bandwidth threshold as the cost of MT1, respectively, the cost of MT2, is lower than the cost of the two trees of the FTS. The cycle is consequently removed. Note that if we have firstly considered the path for d2 instead of d1, we obtain the same result.

4.4. Cycle problem: mQMA vs Mamcra

To compare mQMA to Mamcra on removing cycles from multi-constrained multicast routing structure, we present the following lemmas.

Lemma 1. All cycles removed by mQMA during the tree decomposition step are also removed by Mamcra.

Proof. We have already shown that the tree decomposition step removes cycles when progressively considering the paths of the FTS for grafting. Recall that any path whose multicast members are already considered in the current FTS will not be further considered for grafting. If this path happens to form cycles with the existing trees in the FTS, these cycles are readily dismissed.

In Mamcra, optimal paths are considered in the same order as in mQMA tree decomposition. Before the start of an iteration, Mamcra algorithm removes, from the set of optimal paths, all the paths whose nodes are already visited within the current routing structure. Consequently, as in the tree decomposition, if any of the removed paths happens to form cycles with the current routing structure, these cycles are readily dismissed.

Lemma 2. Some cycles not removed by mQMA may be removed by Mamcra.

Proof. Let us consider the example presented in Fig. 6 with the multicast group having the node s as a source node and nodes d1, d2 and d3 as multicast members, the constraint limits are fixed to (6,6). Both Mamcra and mQMA starts by computing optimal paths to destination nodes which are the path P1 = { s–b–c–d1 }, the path...
Some cycles not removed by Mamcra may be removed by mQMA.

Proof. Let us consider the example presented in Fig. 7 with the multicast group \( g \) having \( s \) as source node and \( d_1, d_2 \) and \( d_3 \) as multicast destinations with the constraint limits fixed to \((10, 10)\).

Using the Samcra algorithm, optimal paths for \( d_1, d_2 \) and \( d_3 \) are, respectively, \( P_1 = (s \rightarrow b \rightarrow c \rightarrow d_1), P_2 = (s \rightarrow a \rightarrow c \rightarrow d_2) \) and \( P_3 = (s \rightarrow b \rightarrow c \rightarrow d_3) \). Mamcra adds first path \( P_1 \) to the routing structure, dismisses path \( P_1 \) and then adds path \( P_2 \). Mamcra fails to remove the cycle \((s \rightarrow a \rightarrow c \rightarrow b)\) by rerouting \( P_2 \) through \( P_1 \) since the path \((s \rightarrow b \rightarrow c \rightarrow d_2)\) has a weight vector \((11, 9)\) which exceeds the constraint limits \((10, 10)\). Mamcra tree decomposition step yield an FTS containing two trees \( FT_1 \) obtained from the path \( P_1 \) and \( FT_2 \) obtained from the path \( P_2 \). Let us suppose that the current MTS contains the tree \( MT_i = [(s \rightarrow a \rightarrow c \rightarrow d_1), (s \rightarrow a \rightarrow c \rightarrow d_2)] \) which has been computed for another multicast group taking into account other requirements. This tree can aggregate our group as it fulfills the constraint limits for all destination nodes and it has a cost lower than the sum of the costs of \( FT_1 \) and \( FT_2 \). The cycle \((s \rightarrow a \rightarrow c \rightarrow b)\) is then removed.

The previous lemmas clearly point out that there are scenarios where mQMA outperforms Mamcra, but there are others where Mamcra outperforms mQMA in terms of the number of cycles removed. The question naturally arises as to the efficiency of mQMA versus Mamcra for the number of cycles removed, but also the performance evaluation of mQMA as a multi-constrained QoS multicast algorithm. These issues are investigated in the following section. □

5. Simulations

This section presents the results of the simulations comparing the algorithm mQMA to the algorithm Mamcra for the number of trees, the optimality according to the multiple additive constraints, the network resource usage and the number of cycles removed by each algorithm. In the first step of mQMA, we have to use a unicast QoS routing algorithm. In the experimental study we adopt Samcra as the unicast QoS routing algorithm to compute the set of feasible paths FPS. Samcra, which is described in Appendix A, is a unicast QoS algorithm that computes multi-constrained optimal path. The optimality is obtained according to the Samcra length function. We adopt this length function to compare mQMA to Samcra for the optimality of the routing structure. Moreover, we compare mQMA and Mamcra for the number of cycles removed from the multi-constrained multicast routing structure (we recall that this routing structure is a set of one or many trees for mQMA and one subgraph for Mamcra. The latter may contain some cycles).

5.1. Parameters of the simulations

The simulations were ran on Abilene network [26] which contains 11 routers and 14 links as depicted in Fig. 8. In Abilene network, 1000 groups were generated randomly with size between 2 and 11 members. The groups were source-oriented and the source for these groups was chosen among four routers, which are known as specific delivering sources. The links \( l \) on the network were given values \( \bar{w}_l \) (for the \( i \)th metric) randomly from 1 to the fixed MaxValueMetric which was 5 during the simulations. We considered 2 additive metrics and we generated the constrained limits \( l_i \) (for the \( i \)th metric) randomly for all the groups and these values where fixed to a maximum value varying from 5 to 50. For mQMA, perfect and leaky match aggregation are adopted and the bandwidth threshold loss \( t_b \) is varied from 0% to 40%, \( t_b = 0\% \) corresponds to the perfect match aggregation. To perform simulations, each simulation case is repeated 1000 times. The simulation tool used to achieve this simulation is based on a graph simulator developed at IRISA and can be found at [27]. This simulator implements graph generating algorithms and several multicast routing and multicast aggregation algorithms.
5.2. Results of the simulations

This section presents the results of the simulations considering the number of trees, the network resource usage, the mean Samcra length per member and the number of cycles removed.

5.2.1. Number of trees

Fig. 9 shows the reduction of the number of trees for mQMA. This performance is due to the multicast tree aggregation. As Mamcra routing structure is not always a tree, in these simulations, the multicast routing structure given by Mamcra is decomposed into trees. These trees must be taken into accounts when counting trees that must be effectively configured in the network. While, in general, Mamcra needs to maintain, at least, as many trees as groups in the domain, i.e., 1000 trees (in the best case, all the multi-constrained multicast routing structure are trees), mQMA reduces the number of trees to only 200. As mQMA builds few trees, few forwarding states are maintained and the control messages to maintain the trees are reduced.

Fig. 10 plots the number of trees for mQMA considering the maximum value of constraints fixed for the groups and a bandwidth threshold loss fixed to 40%. When the constraints are strict and hard to fulfill considering the links metrics values, the number of satisfied members decreases and the trees built often have the same structure. Then, the aggregation can be made more easily. Fig. 11 plots the number of members for which feasible multi-constrained paths are not found and are not considered in the two last steps of mQMA. This number influences the number of trees and even the tree structure as shown in Fig. 10.

The number of tree is also affected by the type of aggregation adopted by mQMA. Fig. 12 shows that the number of trees decreases considerably when leaky aggregation is authorized. Using
these results and based on the bandwidth availability, a network manager may adapt the bandwidth threshold to tune the aggregation rate. This number can reach only one tree if there is an infinite bandwidth threshold: in that case, only one tree covering all the routers of the network is configured.

5.2.2. Network resource usage

Fig. 13 shows the network resource usage in terms of relative utilization. This metric is denoted by

\[ U_R = \frac{C(T_{mQMA}) - C(T_{Mamcra})}{C(T_{Mamcra})} = 1, \]

where \( C(T_{algo}) \) represents the cost of the routing structures of the algorithm \( algo \) used to forward data for all the generated groups. In the simulations, the cost is measured in terms of number of links used for each tree set. \( C(T_{algo}) \) is defined as

\[ C(T_{algo}) = \sum_{g \in \mathcal{G}} |t(g)|. \]

where \( \mathcal{G} \) represents the set of all the groups generated, \( t(g) \) represents the structure used for \( g \) (which can be a set of trees for mQMA or a structure with cycles with Mamcra), and \( |t(g)| \) represents the number of links of \( t(g) \). In mQMA, if the same tree is used to aggregate two groups, it is counted twice.

The relative utilization depends on the tolerated bandwidth threshold. Fig. 13 shows the network resource usage in terms of number of links. The results show that mQMA uses network resources slightly more efficiently than Samcra and Mamcra. This is mainly due to the phase of aggregation. Indeed, mQMA can find an already existing tree that covers the members, that is feasible and that has a lower cost. This occurs when mQMA lists all the trees and chooses among the ones that are feasible, the tree that has the lowest number of links. The results show that even when leaky aggregation is adopted and for lower threshold (≈17%), the aggregated trees used by mQMA to forward multicast traffic have lower cost than the Mamcra routing structure used when no aggregation is performed. So, even when leaky aggregation, mQMA may involve an effective gain in resource usage without violating the multi-constraint requirements of multicast groups. When the threshold increases, the network resource usage is not so high, the network resources waste for an infinite threshold does not exceed 35%.
5.2.3. Mean Samcra length per member

Fig. 14 shows the mean Samcra length for the members, which is the main metric used for multi-constrained routing algorithm. This metric reflects the quality of the communications for the members of the group, expressed in function of the different QoS metrics. The Samcra length from s to d is denoted by

\[ \text{Samcra length}(s,d) = \max_{i \in \text{metrics}} \sum_{e_j \in \text{paths}(s,d)} w_{ij} / L_i, \]

where \( w_{ij} \) is the value of the metric number i for the edge \( e_j \) \((e_j \in P)\) and \( L_i \) is the requirement of the group for the metric i. The Samcra algorithm computes the optimal structure by minimizing the value of this metric. The results show that mQMA behaves in the same way as Samcra with slightly higher results for this metric. Then, we can say that the two algorithms are comparable and that mQMA gives good performance in terms of Samcra length as its results are close to the optimal.

5.2.4. Number of cycles removed

To achieve multi-constrained multicast routing, the routing structure is not always tree. The graph structure computed by the unicast QoS routing algorithm may contain cycles as it is detailed in Section 4. mQMA and Mamcra attempts to solve the cycle problem presented above. Fig. 15 shows performances of the two algorithms to solve this problem. It illustrates, first, the number of cases with cycles contained on the structure computed by the unicast algorithm to reach all the multicast members (continuous line). This represents also the number of cases in which the set FPS of feasible paths contains cycles. Fig. 15 compares the number of cases in which mQMA eliminates cycles to the number of cases in which Mamcra succeed to do it. Recall, that cycles reduction obtained by mQMA, includes the one achieved by the tree decomposition and the one achieved by the aggregation (see Section 1). In some cases, neither Mamcra nor mQMA removes the cycles found in FPS. In most of these cases, the cycles cannot be removed and the structure FPS is the only one possible that respects the constraints fixed for the group.

Table 1 shows the number of cycles removed by Mamcra, mQMA and tree decomposition when generating 30,000 test cases. At the end of the simulations, mQMA has removed slightly more cycles than Mamcra, that means that in 95.59% of cases, Mamcra has removed all the cycles from the FPS while mQMA has removed all the cycles in 96.84% of cases. From these 96.84% cases obtained by mQMA, 68.72% are implied by the tree decomposition step. These cycles are also eliminated by Mamcra as it is stated in (Lemma 1). The other remaining cases are implied by global aggregation, which involves the difference on the number of cycles removed by Mamcra and mQMA as it is stated in Lemma 3. In 73% of the generated cases, the structure FPS found by Samcra contains at least one cycle. This confirms the necessity of finding an algorithm that removes the cycles.

6. Conclusion

In this paper, we detailed mechanisms of the mQMA algorithm that deals with the two main problems of multicast deployment: multicast forwarding state scalability and multi-constrained QoS routing. Our algorithm aims to build aggregated multicast tree under multiple additive constraints. To the best of our knowledge, mQMA is the first protocol that deals with these two main problems. With mQMA, in the worst case, a group can be associated to several trees and the data for the group are multiplexed onto these trees. As multi-constrained multicast routing structure is not always a tree, mQMA contributes to the reduction of this structure by eliminating as many cycle as possible. Our analysis showed that though mQMA is not dedicated to solve the cycles problem, this algorithm has comparable performances to Mamcra algorithm and outperforms it in some scenarios.

Conducted simulations showed that comparable performances to Samcra in terms of Samcra length are achieved. Recall that Samcra provides the optimal routes according to this metric. Despite of adopting in this paper feasible but not necessarily optimal paths, the aggregated trees computed by mQMA are still very close to the optimal unicast routes. Moreover, when perfect aggregation is adopted, mQMA uses less links than Mamcra and consequently utilizes less network resources. Simulations pointed out that even when leaky aggregation is adopted, the additional network resources are not high, yet in some cases, leaky aggregations could even spare network resources.

Furthermore, simulations confirmed the contribution of mQMA to solve the cycle problem and showed that mQMA slightly outper-
forms Mamcra in removing cycles. Finally, mQMA keeps all the benefits of multicast tree aggregation by building very few trees (at maximum 200 trees for 1000 groups for the Abilene network).

Further investigations are underway to evaluate the communication overhead of our centralized proposal, to consider ways to distribute the tree manager and to deal with hierarchical scalability issues.

Appendix A

This section presents Mamcra and Samcra, the QoS routing used to solve respectively multicast and unicast multi-constrained routing problems. First, we give an overview of Samcra as it is used as a basis for the multicast QoS routing algorithm (Mamcra), next, we present, the Mamcra algorithm.

A.1. Samcra algorithm

Samcra returns the path between a given source and a destination node respecting end-to-end constraints and optimizing an objective function. Samcra, like it is described in [24], is based on three fundamental concepts: a non-linear measure for the path length, the k-shortest path approach and the principle of dominated paths. All m additive QoS metrics are equally important. Each link is specified by a m-dimensional weight vector $\mathbf{w} = [w_1, w_2, \ldots, w_m]$. The path vector $\mathbf{w}(P) = [w_1(P), w_2(P), \ldots, w_m(P)]$ is the vector sum of the link weights along this path. The path length is a vector norm and given by $l(P) = \max_{1 \leq i \leq m} (w_i(P)/L_i)$, where $w_i(l) = \sum_{l \in P} w_i(l)$.

This length function obeys the criteria for “distance” in vector algebra and is motivated by the geometry of the constraints surface in m-dimensional space. Samcra algorithm proceeds in a Dijkstra-like manner but by using this length function to explore nodes. Another result of this non-linear function is that the sub-section of shortest paths in a m-dimensional space are not necessarily shortest paths. That is why the k-shortest path approach is adopted and leading authors to consider not only the shortest path like it is done in Dijkstra’s algorithm but also the second shortest, etc., up to the kth shortest path. Non-dominance concept is used to reduce the space search by eliminating dominated paths and exploring only non-dominated ones. Using this concepts Samcra find not only feasible path according to the required constraints but also optimal one according to the non-linear length function defined above.

A.2. Mamcra algorithm

Mamcra proceeds in two steps:

- **Step A**: Compute the set $S$ of shortest paths from source node $s$ to all $p$ multicast members.
- **Step B**: Add paths of $S$ to $M$ and optimize $M$ without violating constraints.

Step A is obtained by applying Samcra algorithm to find the set of paths from $s$ to all destination. The set of paths $S$ may contain overlaps, that is why, the step B attempts to remove some overlap in the set $S$ in order to reduce overlap. This step proceeds in a greedy manner to eliminate overlaps if it is possible. This step can be summarized by the meta-code in Algorithm 3.
Step B: Optimizing delivery structure

**Step (B) of Mammeca**

**Input:** The network $G = (N, E)$, a group $g$ with a source $s$, constraints $L_i$, the set $S$ of optimal path computed by Samcra

**Output:** A set $M$ of paths

While $(S \neq \emptyset)$ do
    add the path with the most members($d_j$) to $M$;
    **If** (many) **then**
    choose the one with smallest length
    **end If**
    **If** (the added path forms a cycle in $M$) **then**
    optimize $M$ by rerouting the new path through an already existing path without violating constraints;
    **end If**
    **If** (cycle is not removed) **then**
    Check if the new path does not violate the min/max constraints
    **end If**
    Remove from $S$ all nodes that are already visited by $M$;
**done**

**return** $(M)$

---

References