SELDOM: A Simple and Efficient Low-cost, Delay-bounded Online Multicasting

Tawfig Alrabiah and Taieb F. Znati
Department of Computer Science
University of Pittsburgh
Pittsburgh, PA 15260
{tawfig, znati}@cs.pitt.edu

Abstract

With the advent of multimedia applications, the support of *on-line* multicasting with quality of service (QoS) guarantees has gained considerable attention in the field of communication networks and distributed systems. The objective of this paper is to investigate on-line QoS-based routing and path establishment schemes to support point-to-multipoint connections in wide area networks. We propose SELDOM, a Simple and Efficient Low-cost Delay-bounded Online Multicasting scheme to support on-line multicasting. The scheme is particularly tailored to networks in which group membership changes frequently.

The approach taken by the scheme is unique in the sense that, given a set of QoS requirement specifications of each multicast node and the current status of the network links, SELDOM finds a minimum cost multicast tree that meets these QoS specifications of the supported group members. The scheme handles *join* requests dynamically by determining the least cost path which satisfies the required delay bounds to which the new group member is to be attached. On the other hand, to handle a *leave* request, the scheme seeks to limit the rearrangement required in order to reduce the disturbance such a request may cause to current members of the group. The worst time complexity of SELDOM is $O(n^2)$.

Keywords

online multicast routing, steiner tree, quality of service

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35388-3_42

H. R. van As (ed.), *High Performance Networking*

1 INTRODUCTION

Recent advances in high-speed networking technology have created opportunities for the development of a wide spectrum of sophisticated multimedia applications which generate, integrate, process, store, and distribute time dependent and time independent media. Typical applications include video conferencing, computer supported collaborative work, and limited video broadcasting. These applications are characterized by a wide spectrum QoS requirements and the need for *group communications* among multiple end-points.

Support of QoS-based group communication in multimedia environments requires the development of efficient and cost-effective *multicast* algorithms. The ability to perform such a task is becoming a major requirement for computer networks that support multimedia applications. To increase the fraction of accepted multicast sessions, the network must use the minimum amount of network resources while guaranteeing the sessions' QoS requirements. From the routing point of view, an efficient multicast algorithm must only replicate packets when necessary, namely at the branching points at the tree.

In the past, the bandwidth required by applications was small and the applications' QoS requirements were not as stringent as those of current multimedia applications. Hence, simple multicast algorithms were used to manage the network resources. However, with the advent of multimedia applications, developing efficient multicast algorithms is becoming increasingly important. To increase the rate of accepted multicast sessions, new algorithms that minimize the amount of replicated traffic exchanged during multimedia multicast session must be developed. These algorithms must guarantee the stringent QoS requirements of multicast sessions while minimizing the cost of the multicast trees used to exchange the resulting traffic.

The multicast group set can be known before setting up the multicast routing tree. In this case, the problem is called the *off-line* multicasting problem. What is required in this case is an algorithm that, given a set of QoS requirement specifications, current status of the network links, and the multicast set, find a *Minimum Cost Multicast Tree* (MCMT) that meets the QoS specification of the multicast tree nodes.

Multicast applications such as teleconferencing, distance learning, collaborative work, and data distribution may require the ability to support dynamic sessions. Dynamic sessions are characterized by their members' ability to join or leave in a dynamic fashion. The sudden and unexpected arrival and departure of session members makes the multicast problem an on-line problem where routing decisions have to be made on-line while the multicast session is in progress. Hence, the multicast group set is not known prior to setting up the multicast session. This problem is called the *Online Minimum Cost Multicast Tree* (OMCMT) problem.

The objective of the multicast problem in a multimedia network is to build a low cost tree that bounds the source-destination delay. That is, given a

graph $G = (V, E)$, where each link is assigned cost and delay, a source node s, a multicast set D, find the lowest cost multicast tree that bounds the source-destination delay. This problem is NP-complete. That can be easily proved by setting up the delay bound to infinitely which reduces the problem to the Steiner tree problem. The Steiner tree problem is a well known NP-Complete problem [19]. Several exact solutions to the Steiner tree problem have been proposed in [8, 19]. All proposed exact solutions, however, require exponential execution time. This prompted the development of several polynomial heuristics for approximate solutions. A survey of these heuristics, as well as exact algorithms, for Steiner problems in networks is provided in [19].

The rest of the paper is organized as follows: we start with reviewing some of the approaches and algorithms proposed to provide an approximate solution to the off-line, low-cost, delay-bounded multicast trees. After that, a discussion of low-cost online multicasting algorithms, which update and maintain multicast trees dynamically in response to join or leave requests, is presented. Section 3 introduces SELDOM which is a new proposed algorithm for the online, low-cost, delay-bounded multicast problem. A conclusion of this work will be presented in the last section.

2 RELATED WORK

Based on their design objectives, the multicast algorithms proposed in the literature can be viewed as members of one of three possible classes. The first class includes algorithms which are designed to accommodate an Internet based environment. The second class includes off-line algorithms which aim at reducing the cost of the multicast tree, while bounding the end-to-end delay. The third class includes algorithms which deal with on-line multicasting. These algorithms are reviewed next.

2.1 IP-based Multicast Protocols

The Internet community proposed different algorithms to create multicast trees, including Distance Vector Multicast Routing Protocol (DVMRP), Multicast Open Shortest Path First (MOSPF), Protocol Independent Multicasting (PIM), and Core Based Trees (CBT) [13, 14, 7, 3]. DVMRP is built on top of RIP (Routing Information Protocol), a distance vector protocol, which is not efficient in detecting loops and link failures quickly. MOSPF uses Open Shortest Path First (OSPF) to maintain a current image of the network topology. CBT builds a single distribution tree, formed around a focal router which is called the core. The major drawback of CBT is the concentration of traffic at the core of the tree. Hence, CBT is vulnerable to core failure which can partition the tree. The Protocol-Independent Multicast (PIM) addresses both dense (PIM-DM) and and sparse (PIM-SM) environments [7]. PIM-DM is en-

visioned to be used in an area where group membership is dense. PIM-DM is similar to DVMRP except the unicast routing information is imported from the existing unicast protocol rather than incorporating it in the implementation of the unicast protocol. PIM-SM creates a center in the tree which is called a Rendezvous Point (RP). Each multicast group has a default router as its RP. New receivers join the tree through the RP. A receiver can switch from the shared tree to a source based tree. Upon switching the source prunes itself from the shared tree.

The above Internet multicast algorithms are designed to work specifically with the current IP environment and to take advantage of the IP routing protocols such as RIP and OSPF. The design objectives of these algorithms focused on issues related to scalability and reduced communication overhead, but did not address the QoS requirements of multimedia applications.

2.2 Off-line, Low-cost, Delay-Bounded Multicast Tree Problem

In addition to low cost, multimedia applications have different demands in terms of bandwidth, reliability, delay and jitter. A key property of multimedia data is its time dependency. The support of sustained streams of multimedia objects, over a period of time, requires the establishment of reliable, low delay and low cost source–to-destinations routes. Nevertheless, the objective is not to develop a strategy which produces the lowest possible end-to-end delay, but a strategy to ensure that the data traffic arrives within its delay bound, thereby allowing a tradeoff between delay and cost. Thus, the objective is to produce a tree that has minimal cost among all possible trees that bound end-to-end traffic delay between all source-destination pairs.

Many heuristics were developed for the low-cost unbounded-delay multicast problem [20, 2]. However, there are few attempts to develop low-cost bounded-delay multicast heuristics. In the following, we review off-line, low-cost, delay-bounded, multicast heuristics. A simple approach to solving this problem is to use a tree that is composed of the least delay paths from the source to the multicast nodes. Such an approach will always find a solution that conforms to the delay bounds if one exists. This approach, however, does not take into consideration any cost optimization. A different heuristic for solving the delay constrained multicast tree is to use the constrained shortest path tree [15]. This heuristic first builds a tree composed of the shortest cost paths to the destinations. If the end-to-end delay to any group member is violated, the least delay path will be used instead.

A dynamic programming approach was suggested by Kompella, Pasquale, and Polyzos [12]. This heuristic assumes that link delays are represented by integer values. The heuristic begins by finding the least cost bounded path from each multicast node to another. Then, it uses the minimum spanning

tree algorithm to connect the multicast nodes without violating the end-to-end delays. The complexity of this approach depends on the granularity of the delay values. If the granularity of the delay is very small then the complexity will be large.

An iterative optimization approach to the minimum delay tree was suggested in [21]. The algorithm starts with the minimum delay tree. Then, it replaces the relay paths with lower cost paths without violating the delay bound. It continues until the cost of the tree cannot be reduced further.

A simple heuristic based on the Simple Path Heuristic (SPH) [16] was proposed in [1]. The heuristic *Least Cost First* (LCF) decouples the cost optimization from bounding the delay by building a low cost tree incrementally and then checking the delay bound requirements. The node with the least cost path to the tree is selected. If the path to that node violates the delay bound, the least-cost delay-bounded path out of the possible low cost paths from the tree to that node is used instead. This process continues iteratively until all multicast nodes are included. Failure to include all multicast nodes implies that no multicast tree which satisfies the QoS requirements for all multicast nodes exists. The analysis shows that the performance and complexity of LCF heuristic are comparable to those of the SPH approximation if the number of delay violations remains moderately small.

Most of the above algorithms were designed for undirected networks. However, they can be implemented in a directed network. Also, they are only designed for static multicast trees (off-line). In the rest of this paper we discuss online multicast tree. All of the online heuristics that we are aware of address low cost online multicasting with no consideration to delay bound. In the following we will discuss some of these online multicasting heuristics.

2.3 Online Heuristic Algorithms

The multicast group membership in typical multimedia settings, such as online video conferencing or multimedia group authoring, dynamically changes as new members request to join the group or current members request to leave the group. Therefore, supporting dynamic multicast applications efficiently requires adding or deleting members to the multicast tree efficiently and transparently to the other multicast members. While many research works have addressed static multicast group communications in WANs, very little research has considered the dynamic version of the multicast communication problem.

An intuitive and trivial solution to this problem is to rebuild the tree using a static algorithm whenever a join request by a new member, or a leave request by a current member, must be handled by the network group management protocol. However, such a solution may have repercussions for members who remain in the group since there may be a disturbance in the communication.

Furthermore, such a change may cause packets to arrive out of order. Another solution is to permit only local or partial reconfigurations when modification to the group membership are required. Yet another approach is to start with an optimal tree and make minimal changes as group membership changes without causing disruption to the members who remain in the group. This approach, however, may not be as efficient as the other approaches because no reconfiguration of the current tree is allowed.

Waxman was one of the first researchers to address the online multicast tree problem [17, 9]. In his work, Waxman partitioned the on-line multicast heuristics into two types: the ones that do not allow rearrangement (*nonrearrangeable*) of the multicast tree and those that allow rearrangement (*rearrangeable*) when the cost exceeds some limit. Several heuristics which approximate the OMCMT problem have been suggested [17, 4, 6, 10, 11, 18, 9], but none of these address supporting the delay requirements of multimedia applications. Following is a review of some of these heuristics.

(a) On-Line Greedy Heuristic (OGH)

This heuristic works as follows. In response to a join request, a node is added to the multicast tree using the shortest path from the current multicast tree to that node. For each leave request, the node is marked as a non-multicast node and is deleted only if it is a leaf node. This is achieved by removing the leaf node from the multicast tree and all branches linking that node to the tree [17]. Imase and Waxman proved that in the case where only node additions are allowed, the worst case cost scenario of the multicast tree produced by OGH is no worse than twice the cost of the multicast tree produced by the best nonrearrangeable algorithm for the online Steiner tree [9].

(b) Edge Bounded Algorithm

Edge Bounded Algorithm (EBA) is a rearrangeable algorithm in which a partial rearrangement is permitted when a modification to the membership occurs. EBA bounds the worst case performance of the generated tree to 4α times that of an optimal Steiner tree, where α is a constant value [9]. Also, it limits the number of rearrangements to $O(K^{3/2})$ where K is the number of (join, leave) requests served.

The algorithm works by creating distance graphs G' and T'. G' is a graph derived from the original graph G. The nodes of G' are those of G. The edges of G', however, are built in a way such that G' is a complete graph. Furthermore, the weights of G's edges are the costs of the minimum cost paths between the nodes of G. A multicast tree T' is created for the node set Z ($Z = \{s\} \cup D$) from G' by pruning the minimum spanning tree of G'. For each join request, EBA selects the least cost path from the new node v to the closest node u in T'. EBA verifies that the added path is α bounded by ensuring that the cost of the maximum cost edge on the path between v and any node u in T' does

not exceed α times the cost of edge (v, u) in G'. If a path to a node u exceeds that limit, v and u will be connected using the least cost path.

Based on this algorithm, a delete request issued by a node v is handled in a way that depends on the degree of v. If the degree of v is one, v is removed using a procedure similar the one described in OGH. If the node has degree three or more, the node will be marked as deleted and no action will be taken. If the degree is two then the node will be removed along with the two adjacent edges. That will create two subtrees. These two subtrees will be connected using an edge that minimizes the cost of the path between the two subtrees.

After serving any join or leave request, EBA verifies that T' is still an extension tree. An extension tree is a tree in which the degree of any non-multicast member is greater than two. If the degree of a non-multicast node is two, the same process used to remove a node whose degree is two is undertaken to build an extension tree.

(c) Shortest Path Tree

Doar and Leslie suggested adding a node by using the shortest path from the source to that node [6]. Thus, the multicast tree will be the union of the minimum source-to-destination shortest paths. Such a tree will give the same result whether the tree was built dynamically or statically. As a result, this approach makes the process of building multicast trees less prone to major spikes of inefficiency. Furthermore, the algorithm does not require handling of rearrangements when nodes join or leave the session. Doar and Leslie showed that such a tree is on average more than 60% worse than the optimal multicast tree. They also suggested imposing a hierarchal model which emulates a real network architecture composed of major backbones and subnetworks. With such a model they showed, by simulation, that the resulting trees are on average less costly than trees produced by non-hierarchal model because there is more sharing of the backbone links.

(d) The Geographic-Spread Dynamic Multicast Heuristic (GSDM)

GSDM is a rearrangement heuristic which was proposed by Kadirire [11]. It is an optimization of the OGH. To illustrate this process, assume that a given node A issued a request to join the multicast tree. Furthermore, assume that node B is the closest tree node to node A, and nodes C and D are the two closest nodes to B. Based on this configuration, the heuristic selects the least cost path among C-B-D & B-A, C-B-A-D, and C-A-B-D. If more than one minimum cost path exists, the heuristic selects the path that maximizes the *Geographic Spread*(GS) of the resulting tree [10].

The GS is defined as follows: let T be a tree that spans Z ($Z = \{s\} \cup D$) where $Z \subseteq V$ and let $v \in V$ and $z \in Z$, the GS is defined as the inverse of the sum of the minimum distance from v to all $z \in Z$ for all nodes $v \in V$ as

shown in equation 1. It has been shown that GSDM heuristic usually performs slightly better than OGH [11].

$$GS(T) = \left[\sum_{v \in V \& z \in T} SP(v, z) \right]^{-1} \quad (1)$$

where SP(v,z) is the shortest path between v and z.

(e) ARIES

ARIES (A Rearrangeable Inexpensive Edge-Based On-Line Steiner Algorithm) is a rearrangeable heuristic [4]. ARIES performs a rearrangement of a region of the multicast tree when the number of modifications (join, leave) within that region reaches a threshold. A region is defined as the part of the multicast tree whose interior nodes are non-stable nodes. A stable node is a node that has never been modified since the start of the tree or the last rearrangement of that region. The performance and the time complexity of ARIES algorithm depend on the threshold value. Small threshold values improve ARIES's performance and increase its run time, and vice versa.

3 HEURISTIC SELDOM

The objective of the OMCMT problem is two fold: minimizing the multicast tree cost and bounding the delays from the source to the destinations. Minimizing the cost by itself is a harder problem since the problem reduces to the Steiner tree problem which is inherently NP-complete. On the other hand, finding a multicast tree which satisfies the specified delay requirements can be solved in polynomial time using one of the classical minimum cost path algorithms [5]. When the two parameters are combined together, the problem is still NP-complete, even in the static case.

On-line multicast algorithms must handle dynamic group membership requests to *join* or *leave* a multicast session in progress. These requests require updating the multicast tree by adding the joining node or removing the leaving node from the tree in a way such that the overall cost of the tree remains minimal and the delay bounds of all multicast nodes are still satisfied. This online version of the multicast problem is still NP-Complete both in a rearrangeable and nonrearrangeable setting. Therefore, some heuristic that gives a "good" approximation with low run time overhead is needed. Furthermore, the heuristic must avoid disrupting the current connections and cause a minimal amount of change to the current connections. More specifically, in the case of joining request, the objective of an online multicast algorithm is to find a low-cost, delay-bounded path to the new joining node, while maintaining the lowest possible number of arrangements. Furthermore, in the case of leave request, the algorithm should delete a leaving node in a way such that the

delay bounds are not violated and the number of arrangements remains low. In the following, we first formulate the online multicast problem. We then propose a new online heuristic, referred to as *Simple and Efficient, Low-Cost, Delay-Bounded Online Multicasting* (SELDOM), which provides an effective solution to handle join and leave requests efficiently.

3.1 Problem Definition

A point-to-point communication network can be represented as a directed graph $G = (V, E)$ where V denotes a set of nodes and E a set of asymmetric links. The network is assumed to be full duplex. In other words, the existence of link $l = (u, v) \in E$ implies the existence of link $l' = (v, u) \in E$. Each link $l \in E$ is assigned a cost value $C : E \longrightarrow \mathcal{R}^+$. A link cost value can be either the link utilization or a monetary value associated with the link. Also, each link $l \in E$ has delay $\delta(l)$, where $\delta(l) \in \mathcal{R}^+$. A link delay may consist of CPU processing, queuing, transmission and propagation. Because network links are often asymmetric, it is often the case that $C(u, v) \neq C(v, u)$ and $\delta(u, v) \neq \delta(v, u)$. If the links are symmetric, then $C(u, v) = C(v, u)$ and $\delta(u, v) = \delta(v, u)$.

A path from node v_1 to v_k is defined as the sequence of nodes and links $P(v_1, v_k) = v_1, v_2, ..., v_k$ such that $(v_i, v_{i+1}) \in E$ for all nodes from v_1 to v_{k-1}. The path $P(v_1, v_k)$ is assumed to be loop free. The cost of a path is the sum of the cost of the links constituting $P(v_1, v_k)$:

$$Cost(P(v_1, v_k)) = \sum_{l \in P(v_1, v_k)} C(l) \tag{2}$$

Similarly, the total delay of a path is the sum of the delay of the links constituting $P(v_1, v_k)$:

$$Delay(P(v_1, v_k)) = \sum_{l \in P(v_1, v_k)} \delta(l) \tag{3}$$

The general MCMT problem can be defined as follows: let the multicast group consist of a source node s and a destination node set D. Given that the maximum delay allowed on the path $P(s, d)$, where $d \in D$, is Δ_d, the MCMT, T, is the tree that satisfies the following two conditions:

$$Cost(T) = \sum_{l \in T} C(l) \qquad \text{is minimum} \tag{4}$$

subject to

$$Delay(P(s,d)) = \sum_{l \in P(s,d)} \delta(l) < \Delta_d \quad \forall d \in D \tag{5}$$

In some cases, the underlying multicast application is delay insensitive. In this case, the delay is not bounded ($Delay(P(s,d)) = \infty$, $\forall\ d \in D$), and the MCMT problem reduces to minizing the total cost of the multicast tree. Furthermore, if the links are undirected, the MCMT is reduced to finding the minimum cost tree, T, that contains all nodes $\{s\} \cup D = Z$. This problem is well known in the literature and is called Steiner Minimal Tree (SMT) [8].

The above definition applies to the case where the multicast nodes are known a priori (i.e. off-line multicasting). However, when the problem is on-line, the multicast tree is dynamic; the multicast nodes in this case may join or leave dynamically. This problem is called the *Online Minimum Cost Multicast Tree* (OMCMT) problem. In this case, a request vector $R = (r_1, r_2, ..., r_k)$ with k requests is given where r_i has three parameters (v, x, Δ_v) such that $v \in D$, $x \in \{add, remove\}$ and Δ_v is the delay bound to that node. The OMCMT tree in this case is defined as the tree that satisfies the two conditions in equations 4, 5 after processing each join or leave request.

3.2 SELDOM Design Approach

Given that some of the networks are not always congested and the number of delay violations may be low, the approach taken by SELDOM to efficiently handle online multicast requests is to first minimize the cost of the paths to destinations, and second verify the feasibility of these paths in supporting the required delay bounds. This approach is being taken since the cost reduction is inherently an NP-complete problem while bounding the delay is a simpler problem which can be performed in polynomial time.

In response to a join request, SELDOM determines the least cost path from the multicast tree to the new node, and verifies the feasibility of the selected path in meeting the delay bound requirements of the joining node. If the delay requirements of the new added node are violated, SELDOM searches for a delay-bounded path with the lowest cost. Following is a description of two modes of SELDOM operations, namely nonrearrangeable SELDOM, and rearrangeable SELDOM.

3.3 Nonrearrangeable SELDOM

In a connection oriented network, it is important to reduce the number of rearrangements of current connections as a multicast tree evolves. This is

especially important when the network is loaded and the network control and routing information is distributed. Rearrangement involves rerouting of information and may require a significant amount of network synchronization and resources. Therefore, a nonrearrangeable online multicast algorithm is desirable when rearrangement of the multicast tree is difficult.

First, the SELDOM nonrearrangeable mode is presented. In this mode, SELDOM does not require any rearrangement of the multicast connections. The nonrearrangeable mode of SELDOM works as follows: for each incoming request, if it is a leave request, the node is marked as a non-multicast node and it is deleted only if it is a leaf node. The deletion is achieved by removing the leaf node from the multicast tree and all branches linking that node to the tree. If it is a join request then the following algorithm is used:

Let G be the network graph, T be the current multicast tree, and v be the node to be added.

1. Find the least cost path SP(T,v) from T to v. If SP(T,v) total delay is bounded then return that path and quit. Otherwise, perform the following steps.
2. Create a new graph G' which consists of G's nodes and G's edges reversed.
3. Let P be the set of the shortest delay paths from v to T's nodes in G'.
4. Remove the edges of T, from G'.
5. Find the set of the least cost paths from v to T's nodes in G' and add them to P.
6. Out of P, pick the path p which satisfies the delay bound such that $delay(p) < \Delta_v$ and $cost(p)$ is minimum.
7. If no such path exists then return "cannot add this node".
8. Return p.

G's links are reversed to speed up the shortest path calculations. The edges of T' are removed to create independent paths. To explain the above idea further and to show the benefit from removing T links, we give the following example. In Figure 1 the source node is node 1 and the current multicast tree includes multicast node 4 (in addition to the source node 1) and the maximum delay bound for all multicast nodes is 7. Initially the multicast tree consists of nodes 1,2, and 4 with link (1,2) and (2,4). The cost of the tree is 2 and delay to node 4 is 4. Assume that a new request comes to add node 5 to the multicast tree. Using SELDOM, it will first try adding node 5 using the least cost path from T to 5 which is path (4,5). However, the delay on that path is 8 which violates the delay bound. Therefore, SELDOM will try to find a better delay-bounded path. First, SELDOM will create a new graph G' which is similar to G but the links are reversed as show in Figure 2. The reversal of the links is performed to speed up the shortest path computation from the violating node to the tree. Then, SELDOM will compute the least delay paths in G from the T nodes to node 5. This can be performed in $O(n^2)$ using the

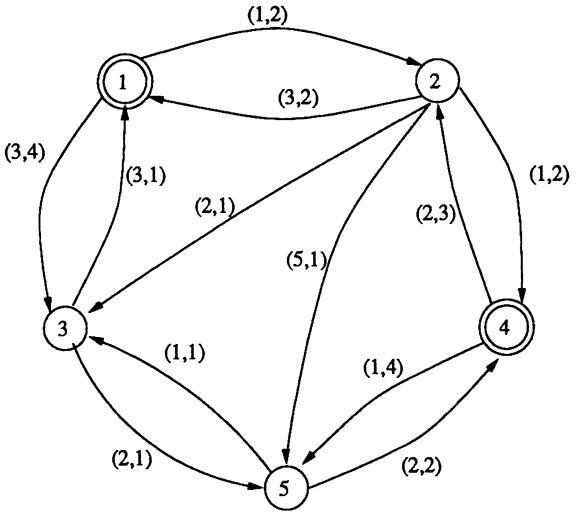

Figure 1 SELDOM example, node 1 is the source node.

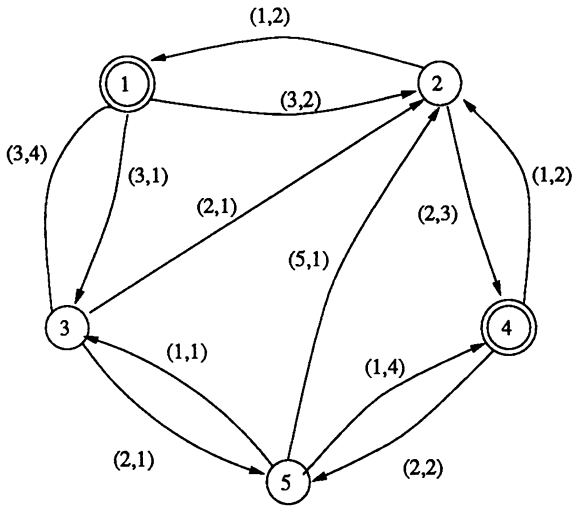

Figure 2 SELDOM example after reversing G's links

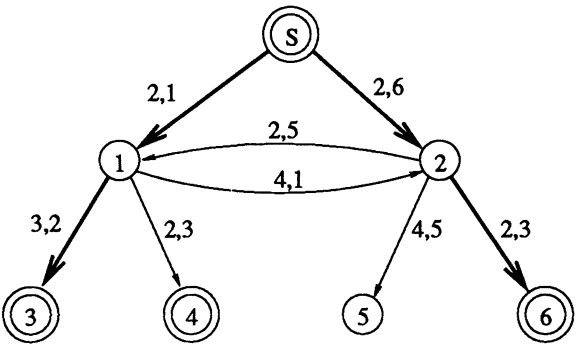

Figure 3 SELDOM example with a source node and two destinations, 2 and 6, each link is assign two values (cost and delay).

shortest delay paths in G'. These paths are: (5,2,1), (5,2), (5,4). After that, the T links ((1,2),(2,4)) are removed from G'. The least cost paths from node 5 to the T nodes will be computed. These paths are: (5,3,1), (5,3,2), and (5,4). Out of these six paths, path (5,3,2) gives the least-cost bounded path with additional cost of 4 and a total delay (from the source to node 5) of 4. If the T links were not removed, the least cost paths will be (5,4,2,1), (5,4,2) and (5,4). These paths are not independent because they use T's links (1,2) and (2,4). Hence, without removing T links, path (5,3,2) would not be discovered.

3.4 Rearrangeable SELDOM

Handling join and leave requests in rearrangeable networks is more involved than in nonrearrangeable settings. In the following, we first describe the operations SELDOM undertakes to respond to a join request. We then describe the operations required to handle a leave request.

(a) Join Request

When adding a node to the current multicast tree in response to a join request, the nonrearrangeable SELDOM may produce a multicast graph which has a node with two incoming paths. As an example, consider the graph depicted in Figure 3. In this graph, the multicast set consists of the source node and two destination nodes, 2 and 6, with a delay bound of 10. The multicast tree in this case is marked using bold links with total cost of 9. Assume that node 5 wants to join the multicast set. In this case, node 5 can be added using the nonrearrangeable SELDOM heuristic. Path 2,5 cannot be used because its traffic will come through link S,2 with a path delay of 11 which violates the delay bound. In this case, node 5 will be added to the multicast tree using

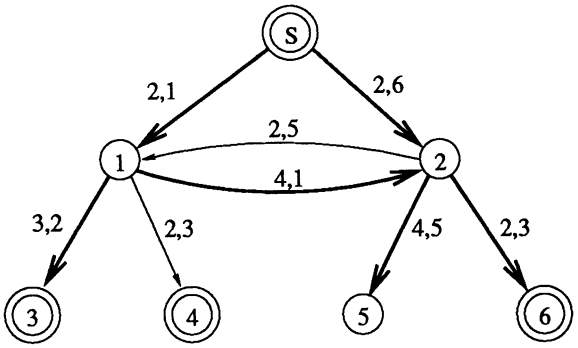

Figure 4 SELDOM example after adding node 5.

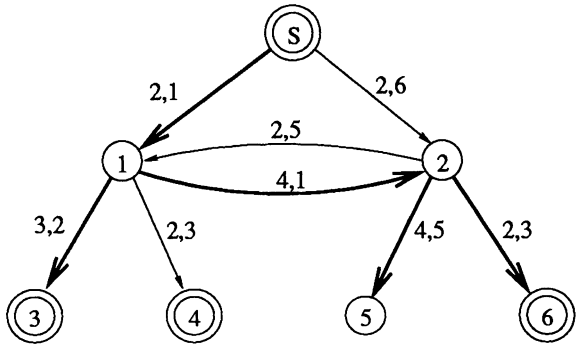

Figure 5 SELDOM example after pruning the link from the source to node 2.

path 1,2,5 as shown in Figure 4 with a total tree cost of 17. This new path makes node 2 have two incoming paths. The generated multicast tree satisfies the delay requirements but this configuration makes node 2 do double work for the same packets.

A multicast tree with a node which has two incoming paths will deliver the multicast traffic as it should. However, that will increase routers' state information. In the above case, a router will have to remember what to do, although it handles the same packets. Also, that will double the workload of the router for the same packets because the router is forced to process the same packets twice. Furthermore, it will increase the cost of tree.

To reduce and possibly eliminate this extra overhead, one of the two incoming paths has to be pruned. Pruning a path, however, should be performed in a way such that no delay bounds to any of the multicast destinations are

violated. In order to achieve this, the path with the larger delay should be pruned. Thus, path S,2 will be pruned because it has a delay of 6 whereas path S,1,2 has a delay of 2 as shown in figure 5. The total cost of the multicast tree after pruning is 15.

Lemma: if the current multicast tree does not have a node with more than one incoming path, then when a node is added, the new path will not create a node with more than two incoming paths.

The above can be proved as follows. When a node is added, the shortest path from a tree node to the new node will not have a cycle because any shortest path cannot include cycles. Therefore, a path will not go through a node more than once. Consequently, the new path will not add more than one extra incoming path to any node in the multicast tree. Since no multicast node has more than one incoming path, the resulting multicast graph will never have a node with more than two incoming paths.

Based on the above we propose a rearrangeable mode of SELDOM. The node join process of this mode tries to reduce the overall cost by pruning the extra paths while minimizing the number of rearrangements. For every node addition it finds all possible paths as it is done in the the previous mode. Then, for each possible path that satisfies the delay bound, it computes the cost of the multicast tree by adding the cost of the new path minus the cost of any possible pruned paths that can exist if that node is added. More formally, the *node-join* algorithm is defined as follows:

1. Find the least cost path SP(T,v) from T to v. If SP(T,v) total delay is bounded then return that path and quit. Otherwise, perform the following steps.
2. Create a new graph G' which consists of G's nodes and G's edges reversed.
3. Let P be the set of the shortest delay paths from v to T's nodes in G'.
4. Remove the edges of T, from G'.
5. Find the set of the least cost paths from v to T's nodes in G' and add them to P.
6. For each possible path in P, compute the cost of resulted multicast tree including the new possible path. The cost of the tree is computed by finding the current cost of the multicast tree including the new path minus the cost of any possible pruned paths that can result from adding that path.
7. Out of P, pick the path p which satisfies the delay bound such that $delay(p) < \Delta_v$ and the total $cost(T)$ is minimum.
8. If no such path exists then return "cannot add this node".
9. Return p.

If all nodes that joined the multicast tree directly use the shortest cost paths without violating any delay bounds, then the produced tree will be similar to the tree produced by OGH (online Greedy Heuristic). It was shown by simulation that when there are only node additions, OGH produces trees

with an average cost that does not exceed that of the optimal tree cost by more than 10%. Imase and Waxman [9] proved that in the worst case the cost of the multicast tree produced by OGH in an undirected graph is no worse than twice the cost the multicast tree produced by the best nonrearrangeable algorithm.

(b) Leave Request

In the nonrearrangeable mode, the deletion of a node in response to a leave request, causes SELDOM to mark the node as a non-multicast node. Furthermore, if the node is a leaf node, all edges and nodes in the relay path linking that node to the tree, including the leaving node itself, are removed from the tree. In the rearrangeable SELDOM, however, node deletion can be improved by performing limited arrangement. The basic steps undertaken by SELDOM to remove a node in response to a leave request are described bellow:

Assume the multicast tree receives a leave request from node v. Let deg(v) be the degree of node v. Also, let a relay path be the path whose all internal nodes have degree two and they are not multicast members. Then the node deletion algorithm can be explained as follows:

If node deg(v) > 2 then
mark v as a non-multicast member.

else if deg(v) $= 2$ then
Delete the relay path from v up to node v_{left} and the relay path up to node v_{right} where v_{left} is the end node of the relay path on the left of v and v_{right} is the end node of the relay path on the right of v. The above will divide the multicast tree into two subtrees.
Assume that the source node is in v_{left} subtree. Also, assume T_1 is v_{right} subtree. Reconnect node v_{left} and T_1 using the least cost path from v_{left} to T_1 which does not violate the delay bound. The least cost path is the lowest-cost, delay-bounded path among the following paths: the least-cost paths from v_{left} to every T_1 node, the least delay paths from v_{left} to every T_1 node, and the old path between v_{left} and v_{right}.

else
Delete v and its relay path up to u where u is the end node of the relay path. If u is not a multicast member then delete u along with its two relay paths up to u_{left} and u_{right}. Assume that the source node is in u_{left} subtree and T_1 is the subtree of u_{right}. Reconnect u_{left} and T_1 in a way similar to connecting v_{left} and T_1 in the above.

The above deletion algorithm limits the number of rearrangements to one. Because it requires finding the shortest path tree, the time complexity of the algorithm is $O(n^2)$. The above enhanced deletion algorithm is expected to reduce the cost of the multicast tree. However, the algorithm can be improved even further. To illustrate this improvement, assume that T_1 is the subtree that includes the end node of the left relay path and T_2 is the the subtree that includes the end node of right rely path. Then, a better path is the one that connects T_1 subtree with T_2 subtree instead of the best path that connects one node with the other subtree. However, that makes the algorithm more complex because it requires finding the least cost paths from every T_1 node to every T_2 node. This process has a time complexity of $O(n^3)$.

Because SELDOM uses the shortest delay paths, it finds a solution if one exists. The time complexity of a node addition is similar to the time complexity of the shortest path algorithm which is $O(n^2)$ where n is the number of nodes in the graph. That is because in the worst case it requires computing the cost of the least-delay paths and the least-cost paths from the multicast nodes to the node being added. By reversing the direction of the links, this still can be done in $O(n^2)$. The complexity of link pruning will never exceed the number of links. Similarly, the time complexity of node deletion is bounded by $O(n^2)$. Hence, the overall time complexity of SELDOM is $O(n^2)$ for each node addition or deletion.

4 CONCLUSION

The multicast problem is NP-complete. Therefore, some heuristics were suggested to give a good approximation with polynomial time complexity. This paper started with a discussion of low cost, delay-bounded multicast trees. Next the online multicast problem was discussed. The online multicast problem is difficult because members join and leave the multicast tree dynamically. One possible solution is to use any of the static multicast heuristics to solve the online multicast problem. However, that will be costly because it requires tearing down and reconnecting the current multicast tree connections. A few heuristics for the online problem were presented. Some of these heuristics did not require rearrangements while the others tried to limit the number of rearrangements. The heuristics that allow rearrangement, however, usually give better results. All of these suggested online heuristics do not bound the delay.

A new heuristic, SELDOM, for online, low-cost multicasting for real-time applications was presented. The nonrearrangeable mode of SELDOM adds a node using the shortest path if that path does not violate the delay bound. If the path violates the delay bound, a search for a low-cost, delay-bounded path is performed. A node is deleted by removing the node and the relay path from the tree to that node. A rearrangeable mode of SELDOM improves the joining process by pruning any possible two incoming paths. Also, it enhances the node leaving process by making limited rearrangement to the graph if that

node has a degree of two. The time complexity of both the nonrearrangeable and the rearrangeable SELDOM is $O(n^2)$.

REFERENCES

[1] Tawfig Alrabiah and Taieb Znati. Low-cost, bounded-delay multicast routing for qos-based networks. *Technical Report*, TR-98-3, November 1997.

[2] Tawfig Alrabiah and Taieb Znati. A simulation framework for the analysis of multicast tree algorithms. *The 30th Annual Simulation Symposium*, 30:196–205, April 1997.

[3] T. Ballardie, P. Francis, and J. Crowcroft. Core-based trees (cgt) an architecture for scalable inter-domain multicast routing. *Computer Communcation Review*, 23(4):85–95, 1993.

[4] F. Bauer and A. Varma. Aries: A rearrangeable inexpensive edge-based on-line steiner algorithm. *IEEE Journal on Selected Areas in Communications*, 15(3):382–397, April 1997.

[5] E. W. Dijkstra. A note on two problems in connection with graphs. *Numeriskche Mathematik*, 1:269–271, 1959.

[6] Matthew Doar and Ian Leslie. How bad is naive multicast routing. *Proceedings of IEEE INFOCOM*, San Francisco, CA:82–93, April 1993.

[7] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, and L. Wei. Protocol independent multicast-sparse mode (PIM-SM): Protocol specification. *Internet RFC 2117*, http://ds.internic.net/rfc/rfc2117.txt, June 1997.

[8] S. L. Hakimi. Steiner's problem in graphs and its implications. *Networks*, 1:113–133, November 1971.

[9] Makoto Imase and Bernard Waxman. Dynamic steiner tree problem. *SIAM Journal of Discrete Math.*, 4(3):369–384, August 1991.

[10] James Kadirire. Minimizing packet copies in multicast routing by exploiting geographic spread. *SIGCOMM Communication Review*, 24(3):47–62, July 1994.

[11] James Kadirire and Graham Knight. Comparison of dynamic multicast routing algorithms for wide-area packet switched (asynchronous transfer mode) networks. *IEEE INFOCOM, Boston*, 19:212–218, April 1995.

[12] V. Kompella, J. Pasquale, and G. Polyzos. Multicast routing for multimedia communication. *IEEE/ACM Transactions on Networking*, 1(3):286–292, June 1993.

[13] J. Moy. Multicast routing extensions for OSPF. *Communications of the ACM*, 37(8):61–66, August 1994.

[14] T. Pusateri. Distance vector multicast routing protocol. *Internet Draft*, February 1997.

[15] Q. Sun and H. Langendoerfer. Efficient multicast routing for delay sen-

sitive applications. *Proceedings of Second Workshop Protocols Multimedia Systems (PROMS'95)*, pages 452–458, April 1995.

[16] H. Takahashi and A. Matsuyama. An approximate solution for the steiner problem in graphs. *Mathematica Japonica*, 24:573–577, 1980.

[17] Bernard M. Waxman. Routing of multipoint connections. *IEEE Journal on Selected Areas in Communications*, 6(9):1611–1622, December 1988.

[18] Bernard M. Waxman. Performance evaluation of multipoint routing algorithms. *IEEE INFOCOM*, pages 980–986, 1993.

[19] Pawel Winter. Steiner problem in networks: A survey. *Networks*, 17:129–167, 1987.

[20] Pawel Winter and J. MacGregor Smith. Path-distance heuristics for the steiner problem in undirected networks. *Algorithmica*, 7:309–327, 1992.

[21] Q. Zhu, Mehrdad Parsa, and J. Garcia-Luna-Aceves. A source-based algorithm for delay-constrained minimum-cost multicasting. *In Proc. IEEE INFOCOM 95*, pages 377–385, 1995.

5 BIOGRAPHY

Tawfig F. Alrabiah received B.A. degree in quantitative methods from King Saud University, Saudi Arabia and the M.S. degree in Computer Science from the University of Pittsburgh in 1995. He also holds an M.S. degree in Information Science from the University of Pittsburgh. He is currently a Ph.D candidate at the University of Pittsburgh majoring in computer Science.

His current research interests are in developing network routing and path establishment schemes to support unicast and multicast communication in distributed multimedia systems. Mr. Alrabiah is a member of ACM and IEEE.

Taieb Znati is an Associate Professor of Computer Science at the University of Pittsburgh, with joint appointments in Telecommunications (Department of Information Science) and Computer Engineering (Department of Electrical Engineering).

Dr. Znati current research interests are in the areas of wired and wireless real-time communication networks, with a particular emphasis on the design and analysis of communication protocols to support distributed multimedia applications. He has published numerous papers in these areas and developed different frameworks to support quality of service requirements of multimedia applications. Dr. Znati chaired multiple conferences and workshops in the field of distributed multimedia systems, high speed communication networks and simulation. He is the current General Chair of the Annual Simulation Symposium, and the General Chair of the Communication Networks and Distributed Systems Modeling and Simulation Conference. He frequently participates in panel discussions about future developments in multimedia systems and high speed networks from the perspective of researchers, developers and users.