Steiner Trees in Industries
D.-Z. Du and X. Cheng (Eds.) pp. – - –

Steiner Tree Based Distributed Multicast Routing in Networks

Roman Novak
Department of Digital Communications and Networks
Jozef Stefan Institute, Ljubljana, Slovenia
E-mail: roman.novak@ijs.si

Joze Rugelj
Faculty of Education
University of Ljubljana, Ljubljana, Slovenia
E-mail: joze.rugelj@uni-lj.si

Gorazd Kandus
Department of Digital Communications and Networks
Jozef Stefan Institute, Ljubljana, Slovenia
E-mail: gorazd.kandus@ijs.si

Contents

1 Introduction

The problem of routing multicast connections in networks is often viewed as a minimum Steiner tree problem in graphs, with additional constraints raised by the specifics of the communication network environments.

Multicasting is a mechanism that allows an efficient distribution of information among a group of collaborating nodes. Emerging applications of communication networks, such as teleconferencing, remote collaboration and distance learning, all make use of multicast communication.

Techniques for transmitting messages to multiple users in a multicast communication session have evolved rapidly over the past few years. Broadcasting is a technique for transmitting a packet from a source node to every other node in a network. Multicasting, i.e. selective broadcasting, is the transmission of a packet from a source node to a limited set of destination nodes, the multicast group.

In order to be efficient, tree-shaped topologies of multicast connections are required. A minimum number of data packets can be transmitted in par-

allel to various destinations along the branches of the tree, with duplication carried out only where the tree branches.

The efficient utilisation of network resources is often considered as the main optimization objective. This is usually true in connection oriented networks while in the networks with best-effort delivery, e.g. MBone in the Internet, the efficient utilisation of network resources is often preceded by the requirement for the shortest delay delivery [9] [2] [8]. In addition to the efficient utilisation of network resources, quality-of-service (QoS) requirements add multiple constraints on the connection's topology. The upper bound on end-to-end delay is an important QoS requirement because most real-time applications, and the interactive ones in particular, are delay-sensitive [27].

Here we pay attention to the most important constraint that affects the design of multicast routing algorithm, i.e. the lack of centralized information about the network topology. This constraint requires an effective distribution of the multicast routing algorithms among the network nodes. A number of centralized multicast routing algorithms exist which take into account different constraints, but only a few distributed multicast routing algorithms have been proposed. Here we survey the distributed multicast routing algorithms known to us. The algorithms solve static or dynamic versions of the multicast problem with or without considering end-to-end delay constraint. Multicast re-routing algorithms are also included.

In Section 2 we define the static and dynamic multicast routing problem. We define end-to-end delay-constrained Steiner trees, and further discuss reasons for distributed implementation and motivation for constructing re-routing algorithms. Section 3 gives a survey of unconstrained distributed static and dynamic algorithms, including our own re-routing algorithm. Distributed constrained multicast routing algorithms are presented in Section 4. Section 5 concludes the survey.

2 Definition of the Multicast Routing Problem

Classification of the multicast routing (MR) problem can be based on several criteria. One may divide the problem into the static and dynamic multicast routing problems, which can be divided further into the unconstrained and constrained versions of the problem. The requirement for distributed implementation of routing algorithms adds significantly to the complexity of the problem. Re-routing multicast connection could gradually deal with this complexity.

2.1 Static MR Problem

In networks that use virtual circuit routing, such as ATM (Asynchronous Transfer Mode) and Frame Relay, a multicast virtual circuit is set up from the source of the multicast to the destinations before data transmission occurs. A virtual circuit is a path between points in a network that appears to be a discrete, physical path but is actually a managed pool of circuit resources from which specific circuits are allocated as needed to meet traffic requirements. ATM networks organizes digital data into 53-byte cell units and transmits them over a virtual circuit, while Frame Relay puts data in a variable-size unit called a frame.

The static MR problem deals with the initial creation of the multicast connection topology. The source of the multicast and the members of the multicast group are known in advance. The multicast connection topology is modelled as a well-known Steiner tree in directed graph [10]. The alternative to the source-based trees is shared trees, where each multicast member node also acts as a source node. The shared tree approach is not generally accepted, because communication links usually have different characteristics in each direction.

Multicast connection topologies that satisfy various QoS constraints have been studied intensively in recent years. Attention has been given mainly to the end-to-end delay bound, which is a prerequisite for real-time communications. Kompella [17] defined the problem of finding a minimum cost tree under the end-to-end delay constraint as the constrained Steiner tree problem. He has also given a proof of its NP-completeness. The constrained Steiner tree problem is formulated as follows.

Definition 2.1 *Given a directed graph $G = (V, E)$ of the communication network with a node set V and an edges set E, we define two non-negative weight functions, $c(e)$ and $d(e)$, on an edge e. $c(e)$ is a cost function on e, and $d(e)$ is a delay function on e. In this graph, we have a source node s and a set of destination nodes D, called the multicast group. A constrained Steiner tree is a tree covering a given subset D of nodes in a graph such that the sum of edges costs in the tree is minimal, while the delay on the path from source s to each destination is bounded above by some given delay tolerance Δ.*

The cost of an edge can be associated with the utilisation of the communication link's resources; a higher utilisation is represented by a higher edge cost. The cost function, $c(e)$, is a function of the amount of traffic traversing

the link e and the expected buffer space needed for that traffic. The value $d(e)$ associated with the link e is the sum of the perceived switching delay, queuing delay, transmission delay, and propagation delay over that link.

Due to NP-completeness of the constrained and unconstrained versions of the Steiner tree problem in graphs, the computation of the optimal solution is not suitable for application in communication network, where fast responses to connection requests are expected. For this reason, Steiner based multicast routing algorithms are heuristic algorithms. A number of centralized multicast routing algorithms exist, but only a subset of Steiner tree heuristics have the properties that make them suitable for distributed implementation in networks where nodes have minimal routing information.

2.2 Dynamic MR Problem

In the dynamic routing problem, multicast members can join or leave the multicast group during the lifetime of the connection. The problem is also known as the on-line multicast problem.

Let $R = \{r_1, r_2, ..., r_k\}$ be a sequence of requests, where r_i is either adding or removing a destination node to or from the multicast group. Let D_i be the set of nodes in the multicast group after request r_i has been made. In response to request r_i, a multicast tree T_i is constructed using a dynamic multicast routing algorithm. The dynamic MR problem can thus be formally defined as follows.

Definition 2.2 *Given graph $G = (V, E)$, a non-negative weight for each $e \in E$, $D \subseteq V$, and a sequence R of requests, find a sequence of multicast trees $\{T_1, T_2, ..., T_k\}$ in which T_i spans D_i and has minimum cost.*

A delay-constrained version of the dynamic multicast routing problem requires the multicast trees T_i to be delay-constrained. Cost and delay metrics are defined for each communication link and the end-to-end delay bound, Δ, is given.

In the extreme case, a multicast tree may be completely rebuilt after each request, using a static multicast routing algorithm. This approach, however, is prohibitively expensive if used for each addition and deletion. Further, real-time multicast sessions cannot tolerate large changes in the multicast tree. This is because data packets are constantly in flight within the multicast tree and large changes to the tree might cause an unacceptable disruption in the packet flow.

2.3 Distributed Versus Centralized Routing

Most of the multicast routing algorithms based on Steiner trees are centralized, i.e. they assume that the information needed about a network is available at one node. While these algorithms are fast and produce near-optimal trees, the requirement that all information be present at one node is problematic in Wide Area Networks (WAN), since the overhead to collect and store the data is prohibitive.

The success of the routing algorithm is based on having an accurate view of the network's state, including its topology and the availability of resources at every node. When a network is small, it is possible to maintain a consistent view of the network state by broadcasting the information needed. As the network gets larger, broadcasting becomes less reliable and more expensive. When the nodes do not have complete knowledge of the topology and state of the network, distributed routing algorithms are needed. Computation of the route would presumably take place on those nodes having the resources, and the resulting decisions would become effective immediately at those nodes. The distributed algorithm collects the information needed and routes the multicast connection in a decentralized manner, with each node carrying out a specified routing algorithm.

Centralized multicast routing is not scalable. A central server should not be used to control multicast routing even when the needed information about network is present at the server. In a large-scale network the server could easily be overloaded, thus degrading network performance. Failure of the central server could easily block the use of the network.

2.4 Multicast Re-Routing

Multimedia flows are relatively long-lived and multicast routing decisions are made only in the connection set-up phase and when nodes join and leave the connection. Every routing decision made may therefore influence the state of the network for a long time [25]. As the state of the network changes over time, the inefficiency of the connection topology tends to grow.

Re-routing enables the use of fast algorithms for building up the initial tree and for adding and removing member nodes. The cost efficiency is gradually improved during the lifetime of the connection. The topology changes must be localized in order to provide smooth transition of ongoing traffic to the alternative paths.

The need for re-routing a static connection topology has already been

recognized. Much work has been done on re-routing unicast connections, but only a few algorithms re-route multicast connections. ARIES [4] makes improvements to the multicast tree while responding to join and leave requests, while our MRR (Multicast Re-Routing algorithm)[23] cannot be classified as either static or dynamic MR algorithm. In addition to MRR, we present here a delay-bounded version of the re-routing algorithm.

3 Unconstrained Distributed MR Algorithms

We restrict ourselves here to the Steiner tree heuristic based approaches that apply to virtual-circuit environments in which low multicast tree cost is of most interest.

During the execution of a distributed routing algorithm the operation of each node should be based on its local routing information. Authors make different assumptions about the available routing information. The availability of a local routing information heavily influences the complexity of the routing algorithm. Many algorithms are not comparable since they are based on different assumptions. Node participation in a distributed algorithm should be limited, as much as possible, to nodes directly involved in the multicast. The time required for establishing a connection must be short.

Competitiveness is often used to compare the quality of trees. The competitiveness is defined as the ratio $(T_{heu} - T_{opt})/T_{opt}$, where T_{heu} is the cost of a multicast tree found by the heuristic and T_{opt} is the cost of an optimal tree. Usually it is given as a percentage.

3.1 Solving the Static MR Problem

Most of the algorithms proposed in the literature for the Steiner problem in networks (SPN) are serial in nature. However, a few distributed algorithms exist.

3.1.1 Minimum Spanning Tree Heuristic (MST)

Many of distributed algorithms for building initial tree are based on reducing SPN to a minimum spanning tree problem (MST) for which a distributed algorithm was proposed back in 1983 by Gallager, Humblett, and Spira [12].

Basically, there are two types of distributed MST algorithm. One type is based on Prim's MST algorithm, where the tree is initialized as the source

node and then grows by successively adding the next closest node to the tree, until all nodes are in the tree. The other type is based on Kruskal's MST algorithm. This type of algorithm initializes each of the nodes as a subtree and joins subtrees pairwise repeatedly until all the nodes are in a single tree.

The distributed MST algorithm in [12] is based on Kruskal's MST algorithm. It starts with each node as a separate fragment. Each fragment executes the algorithm as a sequence of iterations, where each iteration involves finding the minimum weight outgoing edge and combining with the fragment at the other end of the best edge. To determine the best edge, the root of the fragment sends a message to each node in the fragment asking them to report their local best edges. Each node reports this minimum weight outgoing edge back to the root. The information is merged at intermediate nodes as the reports propagate on the way back to the root. Fragment merging is based on level numbers associated with fragments.

The message complexity of the algorithm is $O(|E| + |V|\ log(|V|)$ and the time complexity is $O(|V|\ log(|V|))$. The original distributed MST algorithm has been improved several times [7] [28].

A Steiner tree can be created after an additional phase prunes the minimum spanning tree by removing subtrees without multicast members. There is a big disadvantage to this algorithm, due to all the nodes in the network being involved in the execution of the pruned MST algorithm.

3.1.2 Prim Based Shortest Path Heuristic (P-SPH)

The Shortest Path Heuristic suits well distributed environments, due to its effectiveness and simplicity [31]. Here we refer to it as Prim Based Shortest Path Heuristic (P-SPH) because it grows a single tree. At each step a least cost path is added, from the existing partially built tree to the destination node that has not yet been connected. The heuristic is also known as the Cheapest Insertion Heuristic (CHINS) [32].

Several distributed algorithms have been proposed based on this heuristic. Bauer and Varma [3] treat it as a special case of their distributed Kruskal Based Shortest Path Heuristic (K-SPH) that is described in the next subsection.

Rugelj and Klavzar [26] base their solution on the so-called 'waving' technique. The main problem to be solved in order to implement P-SPH heuristic in network environments is the selection of the closest destination node to the partially built tree, because the least cost path information is

distributed among network nodes. The authors proposed a solution based on 'information collecting' and 'decision-distributing' waves generated in the leaf nodes of the growing Steiner tree and propagated towards its root and vice versa. The wave is a sequence of messages sent across the network connections.

The node where the algorithm was initiated acts as the tree root. Each tree node selects the closest destination node based on its local information. While the selection wave propagates towards the root, the tree node participates with its candidate in the global distributed decision. The root node makes the final selection. The selection is distributed to the tree nodes by the wave in the opposite direction. The closest node to the selected destination node makes a connection and activates the selected destination node as well as all the intermediate nodes on the path to it.

The time complexity of this algorithm, where time is measured as the number of sequentially exchanged messages, is bounded by $O(|D|\ depth(T))$, where $depth(T)$ denotes the depth of a tree. However, the complexity of the cumulative communication time, i.e. time without considering parallel actions, is limited by $O(|Q||V_T|)$, where $|V_T|$ is the number of tree nodes. This also constitutes an upper limit to the number of exchanged messages.

In [24], further optimization of the distribution of the P-SPH heuristic among network nodes is demonstrated. The time needed to select the next closest node can be reduced by enabling nodes to collect the information needed in advance. The distributed algorithm establishes the topology of multicast connection by means of a distributed table-passing technique. The table is a data structure that keeps a record for each destination node not included in the partially built tree. Actually, the table gathers the information about least cost paths from the growing tree to each as yet unconnected group member node. The group member node that was added last decides, according to the information available in the table, which member node should be added next. It then passes the table to the tree node from which the least cost path proposition just selected originates. The selected node establishes a connection with the nearest group member node and passes the table through the intermediate nodes to the newly connected node. The intermediate nodes on the newly created path, including the destination node, update the table according to their local information. If the node has a cheaper proposal for a specific unconnected group member node, it replaces the table record for that node. The distributed algorithm terminates its execution when all the member nodes are connected.

The advantage of the proposed improvement is hidden in the algorithm's

average case behaviour.

3.1.3 Kruskal Based Shortest Path Heuristic (K-SPH)

Bauer and Varma [3] proposed a distributed algorithm that implements the Kruskal based Shortest Path Heuristic (K-SPH). The algorithm uses fragment merging from the distributed MST [12].

Initially, each multicast member node starts out as a fragment. The leader of each fragment attempts to merge with its closest neighbouring fragment. The merge is performed in two steps: a discovery step and a connection step.

In the discovery step, the leader sends a flood message to all nodes in its fragment. On receiving this message, each node sends a query message to neighbouring non-fragment nodes in order to locate nearby fragments. The query message may have to be sent more than one hop to locate other fragments. However, to control the number of query messages, the algorithm limits the distance these messages can travel. Each node reports the fragments encountered to the leader. Based on the information gathered, the fragment leader determines the closest fragment with which to merge.

In the connection phase, the leader sends a request message to the chosen neighbouring fragment. Several scenarios are possible during fragment merging. The chosen fragment may be busy, may reject connection request, or may accept the connection request. After merging, a new leader is selected and the expanded fragment enters into the discovery phase again.

The authors compared the algorithm against the pruned MST algorithm. Its primary advantage is that it requires participation from only those nodes in the multicast tree and in their neighbourhood. The algorithm surpasses the pruned MST in term of competitiveness, but, viewed from the perspective of messages exchanged and convergence time, the pruned MST has the advantage over K-SPH.

The drawback of all Kruskal based approaches is that the algorithms actually build shared trees in networks where links are symmetric. During the execution of a Kruskal based algorithm, the nodes do not have any information about the actual direction of data flow during the multicast session.

In [29] an improvement has been proposed. In particular, the authors avoid sending reject messages as much as possible in order to avoid a new discovery phase. They use an additional flood phase to eliminate incorrect identification of nearby fragments. The number of exchanged messages

reduces from $O(|V||D|)$ to $O(|V|\ log(|D|))$.

3.2 Solving the Dynamic MR Problem

In the dynamic MR problem, nodes join or leave the existing multicast tree. In addition to the Greedy Algorithm (GA) that was originally proposed as a centralized algorithm [33], two distributed algorithms are described: the Predetermined Path Search Algorithm (PPS) [1] and the ARIES (A Rearrangeable Inexpensive Edge-based On-line Steiner Algorithm) [4]. Several other algorithms exist, like Edge-Bounded Algorithm (EBA) [13] and Geographic-Spread Dynamic Multicast Algorithm (GSDM) [16], but only centralized versions of these algorithms are known to us.

3.2.1 Greedy Algorithm (GA)

The Greedy Algorithm (GA) selects the least cost path to an existing multicast tree when adding a node [33]. When GA is implemented in a distributed environment, a newly added node floods the entire network with a query message. The flooding is stopped when the query message reaches either a tree node or a leaf node of the flooding tree. The network overhead can become very large.

For each delete request, GA deletes only leaf nodes. If this deletion creates a nonmember leaf, GA also deletes the new leaf. This continues until no nonmember leaves remain.

The lower bound on the competitive ratio of the greedy algorithm was derived as $2^{|D|}$.

3.2.2 Predetermined Path Search Algorithm (PPS)

The Predetermined Path Search Algorithm (PPS) is based on the greedy algorithm [1]. Two kinds of spanning trees are prepared for the predetermined paths: a minimum spanning tree (MST), and a shortest path tree (SPT) having a multicast source node as a root. These two trees have to be computed in advance using arbitrary algorithms. The PPS algorithm works as follows. A newly added node sends query messages to nodes on two predetermined paths toward the source. If a queried node is a tree node, it sends a reply message to the newly added node and the query message is not transmitted to the next node. Otherwise, the query message is transmitted to the next node on the predetermined path. After the added node

receives reply messages from existing nodes on each predetermined path, it connects to the closest node among those that sent reply messages.

For the PPS algorithm, the competitive ratio is $max(|V|-|D|, |D|)$ since the worst case is always bounded by either the pruned MST or the pruned SPT.

3.2.3 ARIES Algorithm

The upper bound for the competitiveness of the greedy algorithm exists when only add requests are honoured. Imase and Waxman have shown that no such finite bound exists if delete requests are also honoured [13]. This was the motivation for developing heuristic ARIES (A Rearrangable Inexpensive Edge-based Steiner algorithm) [4].

For each add request, ARIES joins the new member to the existing tree by its least cost path to the tree. In addition, ARIES defines the rearrangement mechanism. When the accumulated damage to a part of the tree reaches a certain threshold, ARIES rearranges that region of the tree. Damage to the tree is monitored by the number of changes in the immediate vicinity of member nodes. Each member node has one counter corresponding to each of its edges that are part of the tree. When a member node is deleted or a new node added, the node sends a message to all its multicast neighbours to increase their counters by one. Propagation of the counter-update message is restricted to regions within the tree. A rearrangement is triggered when the value of a counter in a multicast node exceeds a chosen threshold. The rearrangement is performed by the distributed K-SPH.

The algorithm may be tuned to achieve the desired trade-off between performance and cost. An upper bound for the competitiveness of ARIES, considering both add and delete requests, is given by the counter threshold that triggers a rearrangement.

3.3 Multicast Re-Routing Algorithm (MRR)

Re-routing changes the topology of the connection during its lifetime. For this reason, it does not solve static or dynamic MR problems. Routing algorithms must exist in the network to solve both types of problems, in addition to re-routing algorithms. However, the existing multicast routing algorithms may be based on faster and less cost efficient Steiner tree heuristics.

The foundation of the re-routing algorithm is the Steiner tree improvement procedure [32]. In particular, the Multicast Re-Routing algorithm

(MRR) [23] is based on an edge-oriented transformation. The transformation is performed as follows. A simple path is selected randomly in the tree and deleted from the tree. The simple path is defined as a path in the tree of which all intermediate nodes are non-member nodes of degree two with respect to the tree, and whose end nodes are member nodes or non-member nodes of degree three or more. The resulting two disconnected components of the tree are reconnected by a least cost path.

The edge-oriented transformation is very general, because it assumes no restriction on the location of the alternative path. Switching between the original simple path and the alternative path can cause a disruption of the multicast service offered to the clients. In order to cope with this problem efficiently and to simplify distribution of the algorithm to network nodes, two restrictions have been introduced on the selection of the alternative path. The allowed alternative path is the one that has an end node in common with the simple path at the one end, while at the other end, the alternative path should be terminated at the tree node that is in close neighbourhood of the node where the original simple path has been terminated.

The distributed re-routing algorithm consists of three steps: advertising the end node of the simple path, interrogating the set of candidate nodes, and the transition to the alternative paths.

During the advertising of the end node of the simple path, end nodes periodically send the end node reports on each simple path to the other end node. Intermediate nodes update the end node report by summing the reported cost and the cost of the next outgoing link and then forward the report. On receiving the end node report, the other end node associates the reported end node and the reported cost with the outgoing connection from which the report has been received.

Each end node gathers the information on possible alternative paths during the interrogation of the set of candidate nodes. The second step can be initiated in every end node in the tree. It is triggered periodically. The end node distributes enquiry messages to the neighbourhood in order to determine the best candidate for the other end node of potential alternative paths. The distribution is similar to multicast data forwarding, with the exception that at a given hop distance the distribution is terminated. The border nodes return the enquiry messages back to the originating end node. The least cost alternative path is selected during the enquiry for a simple path originating at the end node.

The distributed algorithm allows many re-routing processes to proceed in parallel. However, coordination is needed while re-routing processes are

taking place close to each other. The transition to the alternative path is based on a locking mechanism. The locked node should delay or refuse all requests that result in a tree topology change before it is unlocked. First, the original simple path is locked by sending a lock message. Then, the alternative path is locked. Finally, the distributed algorithm locks the tree path between the end node of the alternative path and the end node of the simple path. If the locking succeeds, a message is sent back along the locked path. The message unlocks nodes, activates nodes on the alternative path, and removes the original path from the tree.

In [23] the costs of the stable trees, of trees obtained by the Shortest Path Heuristic, and of the initial trees are compared. The initial trees were simply the union of the least cost paths between the source node and each of the member nodes. The cost effectiveness of the re-routing algorithm depends on the size of the neighbourhood. For neighbourhoods with 5 simple paths in diameter the re-routing algorithm comes within 10% of the sub-optimal solution.

4 Constrained Distributed MR Algorithms

4.1 Solving the Static MR Problem

A recent survey of mainly centralized delay-bounded routing algorithms for building an initial tree and their evaluation can be found in [27]. Here we survey only distributed algorithms.

4.1.1 Shortest Path Tree (SPT)

The SPT builds a delay-bounded multicast tree by joining shortest delay paths that connect the source to each destination. The algorithm is a distributed version of the Bellman-Ford shortest path algorithm [5]. The tree has to be pruned to remove unnecessary branches. It is guaranteed to succeed in finding a constrained spanning tree if one exists, but does not attempt to optimize the cost of the solution.

4.1.2 Kompella's DMCTs

Kompella et al. proposed two distributed delay-constrained algorithms [18]. Both are based on a distributed minimum spanning tree (MST) algorithm. They mimic Prim's MST algorithm, as the tree is built from a single node.

The two algorithms differ in their criteria for choosing edges while constructing the MST. The selection function f_C selects the cheapest outgoing edge from the subtree constructed so far. A second selection function f_{CD} takes into account the fact that delay also plays a prominent role. The cost of an edge is divided by the residual delay of the path from the source to a given node in the subtree. The residual delay at node v is defined as difference between Δ and the delay on the path from the source node to v in the tree.

It takes one round trip along the tree formed so far to select an edge and add it to the tree. A find message is first sent from the source node to the tree leaves, node by node, to determine the best outgoing edge at each node, using one of the selection functions; then the best outgoing edge is propagated from the leaves back to source, replaced by a better choice along the way. The best outgoing edge should not produce a cycle, and at least one destination node should be reachable within the delay bound.

This algorithm will not always produce a delay-constrained tree, when one does exist. The problem is resolved by the additional cycle make-and-break phase. This phase requires an extra set of messages to be sent from the source to each node in the tree. Each time a cycle is made and broken, the number of unreachable destinations decreases by one. In the end, the tree should be pruned of unnecessary edges. Kompella et al. have shown that their algorithms run in time $O(|V|^3)$.

Low and Lee integrated an additional constraint in DMCT using the f_C as a selection function [19]. In addition to end-to-end delay bound, they considered an inter-destination delay. They require the multicast tree to guarantee a bound on the variation among the delays along the individual source-destination paths.

4.1.3 Delay-Bounded MR Algorithm of Jia et al.

The algorithm is based on a variant of P-SPH, where end-to-end delay is considered at each step [15] [14]. The algorithm builds a single tree. The construction of a routing tree starts with a tree containing only source node. A member node, which is closest to the tree under delay constraint, is selected and the least cost path from the tree to this destination is added into the tree. This operation repeats until all multicast member nodes are in the tree.

The authors of the algorithm assume that each node in the network has the information about the least cost path and the delay of the path to every other node in the network. In addition, they assume that the least cost path

between two nodes is always the shortest delay path between them. This can be true only in networks where the cost of the bandwidth consumption and the delay of a link are in proportion to the distance of the link. The last assumption simplifies the algorithm design. It is easy to check whether the least cost path satisfies the delay bound, because the information needed is present at every node.

The distributed algorithm builds a tree in a way that is similar to that in the version of the unconstrained distributed P-SPH [23]. To record the cost from the tree to the multicast member nodes as the tree grows, a table is introduced. An entry exists in the table for each member node. Each entry has three fields, representing the cost from the tree to the member node, the tree node closest to the member node, and the status showing if a member node is in the tree or not. Each tree node needs to remember the accumulated delay from source node to itself along the tree. The table is initialized at source node and carried in a connection setup message from node to node during routing. It is updated as the message passes through each new tree node.

The source node selects the destination closest to the tree and sends a connection setup message to the neighbour through which the selected destination can be reached by the shortest path. This setup message carries the table and the delay from the source. The intermediate node records the delay and updates the table. Then the setup message is sent to the next neighbour leading to the designated destination. When the message reaches the designated destination, a nontree destination which is the closest to the tree is selected as the next one to be linked in the tree. A fork message is then sent to the tree node that is the closest to the selected nontree destination. On receipt of this fork message, the tree node continues the connection setup by sending down a setup message via its neighbour to the newly selected destination.

The number of exchanged messages and the convergence time is $2|D|$, where the transmission of a control message from a source node to a destination node is counted as the passage of one message.

Rather stringent assumptions regarding the relation between cost and delay metrics are not present in two of our earlier proposals. In [20] only least cost paths to other nodes and delays on outgoing links are known to network nodes, while in [21] the opposite assumptions have been made. The network nodes possess the information about shortest delay paths to other nodes and costs of outgoing links. Both algorithms are based on the similar idea of table passing mechanisms with additional cycle make-and-

brake steps. It has been shown that both algorithms have better average case behaviour regarding the tree cost and the algorithm convergence time than Kompella's DMCTs.

4.2 Solving the Dynamic MR Problem

4.2.1 Dynamic Algorithm of Jia et al.

Jia et al. [15] [14] extend their delay-bounded static algorithm to cover join and leave requests from member nodes.

When a member node requests to leave a multicast group, it is disconnected from the connection. A leave request is sent upward along the routing tree, node by node, until it reaches a fork node. From each node through which this request passes, the connection is released.

When a node wants to join an existing multicast group, it sends a request to the source of the group. The source then sends a join request to the current multicast members. The request contains two fields in order to calculate the cost and originating node of the least cost delay-bounded path from the tree to the new destination. As the request is distributed along the tree connections, each node modifies the least cost proposition if it has a less costly path to the new member under the delay constraint. When the request arrives at a leaf node along the tree, the leaf node sends the gathered information back to the source. After collecting replies from all the leaf nodes of the tree, the source knows the tree node closest to the new member node and the cost from the tree to it. It then sends a fork message to this tree node to extend the connection to the new destination.

Due to the assumption that the least cost path is always the shortest delay path, no cycles occur in the multicast connection topology if the routing information in network nodes is stable. Otherwise, the routing algorithm simply terminates and the connection request is rejected.

4.2.2 WAVE Algorithm

The WAVE algorithm does not consider explicitly a given delay constraint, although it considers more than one link metric [6]. It can be easily adapted to support delay-bounded dynamic multicast in distributed fashion.

WAVE works as follows: when a new member joins an existing tree, it contacts the source node with a request. Starting from the source node, this request is propagated throughout the existing tree. When a node on the tree receives a request message, it sends a response message to the new

member node along the shortest delay path, where the response message contains the cost and delay of the connection from the source node to the new member node via the node that generated this response message. The new member node receives a set of responses from which it selects one and then sends a connection request to the node that generated this response. The selected node extends the existing tree from itself to the member node along the shortest delay path.

This algorithm, as well as the algorithm by Jia et al., would not scale well because the source node is always contacted whenever a new node is added and the number of responses could be very large.

4.2.3 Sun's DCDMR

The distributed Delay-Constrained Dynamic Multicast Routing algorithm (DCDMR) requires network nodes to have the information about costs and delays of least cost paths and, at the same time, the information about costs and delays of shortest delay paths [30]. The algorithm is based on a distributed Delay-Constrained unicast Routing algorithm (DCR).

DCR constructs a delay-bounded path between two nodes. A path-construction message passes through nodes along the constructed path. Each node decides whether to forward the message on the least cost path or the shortest delay path. If the message has arrived on the least cost path then the node forwards it on the least cost path, otherwise it checks the sum of the accumulated delay of traversed links and the delay on the least cost path. If this sum is lower than the delay bound, the next node on the least cost path is selected, otherwise the next node on the shortest delay path receives the path-construction message. The worst case message complexity of DCR is $O(|V|)$.

DCDMR can work in two different modes, namely the fast mode or the slow mode. When a new member joins an existing tree, it contacts a node on the existing tree along the shortest delay path. In the fast mode the DCDMR algorithm takes the following steps. If the sum of the delay along tree edges from the source to the contacted node and the delay on the shortest delay path toward the new member node is less than the delay bound, the contacted node computes a path from itself to the new member using DCR. Otherwise, the contacted node sends the received request message to the node's parent node on the existing tree. The parent runs the same algorithm. If the request message is received by the source node and the delay on the shortest delay path is not less than delay bound then no delay-constrained

path exists and DCDMR terminates.

In the slow mode further optimization is performed. If the delay in the contacted node cannot be satisfied, the DCDMR acts as in the fast mode. Otherwise, the contacted node sends the cost of the least cost path between itself and the member node to its parent and children if the delay on the least cost path falls within bounds. If that is not true, it sends the cost of the shortest delay path. The receiving node checks if the delay bound from itself can be satisfied. If not, it sends the received message to the member node along the shortest delay path. Otherwise, the node updates the received message if it has a better cost proposition and distributes the message up or down the tree.

The new member node receives a set of responses from which it selects the one with the lowest cost value. The member node then sends a connection request to the node that generated the best proposition. The selected node computes a path from itself to the member node using DCR.

Compared with the WAVE algorithm, the number of responses sent to the new node in the slow mode could be much smaller. If DCDMR works in the fast mode, even less computation at tree nodes is needed. The worst case message complexity for the slow mode is $O(|V|^2)$ and the message complexity for the fast mode is $O(|V|)$. The average cost performance has been evaluated in comparison to the Bounded Shortest Multicast Algorithm (BSMA) [34], which is the best known delay-constrained static multicast heuristic. The DCDMR in the slow mode has a cost performance that is 10% or less worse than BSMA, while the DCDMR in the fast mode is up to 40% worse than BSMA.

4.3 Delay-Constrained MRR Algorithm (DCMRR)

The MRR algorithm [23] can be extended in order to re-route delay-bounded multicast trees. We have already presented two modifications of the edge-oriented transformation that are necessary for its application to distributed multicast re-routing. In addition, a delay bound Δ should be incorporated in the modified edge-oriented transformation, so that the tree modified by the transformation will satisfy given constraints.

In [22], we assumed that each tree node knows the allowed delay for each outgoing link. The allowed delay is the sum of actual delay and excess delay. The excess delay must be assigned to the link during the tree construction process. Distribution of total excess delay is part of several strategies for providing quality-of-service guarantees. The sum of allowed

delays on each path from the source node to destination node should never exceed Δ. The alternative path should be picked in such a manner that this condition remains satisfied.

The allowed delay for the alternative path can be computed only by knowing the allowed delay values on the edges of the original simple path and on the tree edges between the end nodes of the simple and alternative path. If the actual delay on the alternative path is less or equal to the allowed delay for the alternative path, the excess delays can be computed for the alternative path in such a way that the resulting tree still satisfies the delay constraint.

The nodes are required only to be aware of the costs and delays on all adjacent edges, and the next node on the path towards any other node in the network. Two phases of the distributed algorithm can be recognized. The first phase implements the search for a simple and alternative path, while the second phase performs the transition from the simple path to the alternative path.

Re-routing is initiated periodically in the end node of each simple path. The end node sends a search message on the simple path towards the source node. The message is distributed only to a limited set of nodes. The limit is set on the number of traversed simple paths. At each node the message also bounces towards the originator of the search. If the search message manages to travel all the way to the originating end node then the traversed path is a potential alternative path. While the search message is forwarded, the necessary computations are performed by the traversed nodes in order to check delay bound and loop-free topology.

The second phase is activated by the end node after it has been informed about the simple and alternative paths during the last search phase. The locking mechanism is implemented because many re-routing processes can take place close to one another. The objective of locking is to prevent parallel transitions from corrupting the multicast tree topology and, at the same time, to verify consistency of the information about the alternative path. A lock message traverses the circle made by the simple path, the tree path between the end nodes of the simple path and of the alternative path and the alternative path itself. The message completes the circle only if there are no previously locked nodes detected and if there is no inconsistency with the information gathered in the search phase. After the successful locking, the alternative path is established and the original path is removed from the tree.

We performed extensive analysis of the distributed DCMRR algorithm.

Among others, we compared the cost of the tree obtained by the re-routing with the cost of the initial tree obtained by the union of shortest delay paths. The maximum improvement in the tree cost reaches 60%. The multicast tree cost can be reduced significantly, especially when group members are distributed sparsely across a wide area. In that case, the cost of stable trees comes within 10% of the optimal cost.

5 Conclusion

Steiner trees can be recognized in the topology of multicast connections in communication networks when cost optimization is of primary interest. In the paper a subset of known multicast routing algorithms is considered. All the algorithms are based on the common assumption that the information about network topology is not present at one node but it is distributed among network nodes.

Three types of MR algorithms have been considered. Static MR algorithms are used when a group of destination nodes is known in advance. Dynamic MR algorithms deal with cases when destination nodes join or leave the multicast group during the lifetime of the connection. The third type of MR algorithms are re-routing algorithms. They constantly optimize multicast connection topology in response to changes of the network state.

The constraint Steiner tree problem was defined in order to establish multicast connections that satisfy various quality-of-service constraints. Attention has been paid mainly to the delay bound. We divide MR algorithms into those that satisfy additional end-to-end delay constraint and those that do not.

Due to NP-completeness of the Steiner tree problem, heuristic algorithms have been adapted for multicast routing. Only a few of the known heuristics are suitable for distributed implementation. Minimum spanning tree heuristic (MST) and shortest path heuristic (SPH) are the most frequently considered for the static MR algorithms. SPH has advantage over MST as regards cost competitiveness and because fewer network nodes are required to be involved in the algorithm execution. Re-routing algorithms are based on the Steiner tree improvement procedure. In particular, modified edge-based transformation has been used.

Two techniques for gathering distributed information about network topology can be identified. Most of the algorithms use the waving technique, where addition of the next node requires a substantial exchange of

messages between network nodes. The table based technique collects the information for multiple decisions in the table. Fewer messages have to be exchanged but their length increases. The table based algorithms have been proposed by Jia et al. and by our group.

The distributed algorithms are not comparable due to the different assumptions they make about the information available at network nodes. The unconstrained algorithms require from each node at least knowledge of the information about the costs of its outgoing links (pruned MST). Most of them also need the costs of least cost paths and the next node on these paths to be available at each node. For instance, the PPS algorithm even assumes the existence of a predetermined shortest path tree and minimum spanning tree. Among the constrained MR algorithms, DCMRR needs only the costs and delays of outgoing links and the next node on an arbitrary path towards other nodes to be available at each node. Jia et al. require the knowledge of the costs and delays of least cost paths and, at the same time, least cost paths to be always shortest delay paths. The costs and delays of least cost paths and the costs and delays of shortest delay paths are assumed by Sun. The other assumptions include the costs of least cost paths and the delays of outgoing links or vice versa, the delays of shortest delay paths and the costs of outgoing links, in addition to the next node on the path of given type towards other nodes in the network.

Dynamic network environments, distributed information about network topology and state, and the complexity of the Steiner tree problem favour development of efficient multicast re-routing algorithms. Static MR algorithms as well as dynamic MR algorithms are of limited efficiency as their solutions are optimal only for a limited period. The need for re-routing unicast connections has been already recognized in the past while re-routing multicast connections is still a relatively new subject.

References

[1] T. Asaka, T. Miyoshi, and Y. Tanaka, New distributed multicast routing and its performance evaluation, *IFIP Networking 2000*, (Paris, France, May 2000).

[2] T. Ballardie, P. Francis, and J. Crowcroft, Core-based trees (CBT): An architecture for scalable inter-domain multicast routing, *Computer Communications Rev.* Vol.23 No.4 (1993) pp. 85-95.

[3] F. Bauer, and A. Varma, Distributed algorithms for multicast path setup in data networks, *IEEE/ACM Transactions on Networking* Vol.4 No.2 (1996) pp. 181-191.

[4] F. Bauer, and A. Varma, ARIES: A rearrangeable inexpensive edge-based on-line Steiner algorithm, *IEEE Journal on Selected Areas in Communications* Vol.15 No.3 (1997) pp. 382-397.

[5] R.E. Bellman, *Dynamic programming*, (Princeton University Press, Princeton, N.J., 1957).

[6] E. Biersack, and J. Nonnenmacher, WAVE: A new multicast routing algorithm for static and dynamic multicast groups, *Proceedings of 5th Workshop on Network and Operating System Support for Digital Audio and Video* (Springer Verlag, Heidelberg, Germany, 1995) pp. 228-239.

[7] G. Chen, M. Houle, and M. Kuo, The Steiner problem in distributed computing systems, *Information Sciences* Vol.74 No.1-2 (1993) pp. 73-96.

[8] S. Deering et al., The PIM architecture for wide-area multicast routing, *IEEE/ACM Transactions on Networking* Vol.4 No.2 (1996) pp. 153-162.

[9] S. Deering, and D. Cheriton, Multicast routing in datagram internetworks and extended LANs, *ACM Transactions on Computer Systems* Vol.8 No.2 (1990) pp. 85-110.

[10] C. Diot, W. Dabbous, and J. Crowcroft, Multipoint communication: a survey of protocols, functions, and mechanisms, *IEEE Journal on Selected Areas in Communications* Vol.15 No.3 (1997) pp. 277-290.

[11] J. Ford, and D.R. Fulkerson, *Flows in Networks*, (Princeton University Press, Princeton, N.J., 1962).

[12] R.G. Gallager, P.A. Humblett, and P.M. Spira, A distributed algorithm for minimum-weight spanning trees, *ACM Transactions on Programming Languages and Systems* Vol.5 No.1 (1983) pp. 66-77.

[13] M. Imase, and B. Waxman, Dynamic Steiner tree problem, *SIAM Journal on Discrete Mathematics* Vol.4 No.3 (1991) pp. 369-384.

[14] X. Jia, A distributed algorithm of delay-bounded multicast routing for multimedia applications in wide area networks, *IEEE/ACM Transactions on Networking* Vol.6 No.6 (1998) pp. 828-837.

[15] X. Jia, Y. Zhang, N. Pissinou, and K. Makki, A distributed multicast routing protocol for real-time multicast applications, *Computer Networks* Vol.31 No.1-2 (1999) pp. 101-110.

[16] J. Kadirire, Comparison of dynamic multicast routing algorithms for wide-area packet switched (Asynchronous Transfer Mode) networks, *Proceedings of IEEE INFOCOM'95* (Boston, MA, April 1995) pp. 212-219.

[17] V.P. Kompella, J.C. Pasquale, and G.C. Polyzos, Multicast routing for multimedia communication, *IEEE/ACM Transactions on Networking* Vol.1 No.3 (1993) pp. 286-292.

[18] V.P. Kompella, J.C. Pasquale, and G.C. Polyzos, Optimal multicast routing with quality of service constraints, *Journal of Network and Systems Management* Vol.4 No.2 (1996) pp. 107-131.

[19] C.P. Low, and Y.J. Lee, Distributed multicast routing, with end-to-end delay and delay variation constraints, *Computer Communications* No.23 (2000) pp. 848-862.

[20] R. Novak, and J. Rugelj, Distribution of constrained Steiner tree computation in point-to-point networks, *Proceedings of the Fourteenth IASTED International Conference on Applied Informatics* (Insbruck, Austria, February 1996) pp. 279-286.

[21] R. Novak, and J. Rugelj, Distribution of constrained Steiner tree computation in shortest delay networks, *Proceedings of the 8th Mediterranean Electrotechnical Conference on Industrial Applications in Power Systems, Computer Science and Telecommunications* (Bari, Italy, May 1996) pp. 959-962.

[22] R. Novak, and J. Rugelj, Rerouting of multicast connections with quality-of-service constraints, *Proceedings of the 23rd Euromicro Conference* (Budapest, Hungary, September 1997) pp. 301-307.

[23] R. Novak, J. Rugelj, and G. Kandus, Re-routing multicast connections: a distributed approach, *Journal of Computing and Information Technology* Vol.7 No.4 (1999) pp. 323-331.

[24] R. Novak, J. Rugelj, and G. Kandus, A note on distributed multicast routing in point-to-point networks, *Computers & Operations Research* (2001).

[25] C. Parris, H. Zhang, Dynamic re-routing of guaranteed quality-of-service connections, *Journal of Network and Systems Management* Vol.4 No.2 (1996) pp. 181-220.

[26] J. Rugelj, and S. Klavzar, Distributed multicast routing in point-to-point networks, *Computers & Operations Research* Vol.24 No.6 (1997) pp. 521-527.

[27] H.F. Salama, D.S. Reeves, and Y. Viniotis, Evaluation of multicast routing algorithms for real-time communication on high-speed networks, *IEEE Journal on Selected Areas of Communications* Vol.15 No.3 (1997) pp. 332-345.

[28] G.Singh, and A. Bernstein, A highly asynchronous algorithm for minimum spanning tree, *Distributed Computing* Vol.8 No.3 (1995).

[29] G. Singh, and K. Vellanki, A distributed protocol for constructing multicast trees, *Proceedings of International Conference on Principles of Distributed Systems*, (Amiens, France, December 1998).

[30] Q. Sun, and H. Langendörfer, A distributed delay-constrained dynamic multicast routing algorithm, *Telecommunications Systems Journal* Vol.11 No.1-2 (1999) pp. 47-58.

[31] H. Takahashi, and A. Matsuyama, An approximate solution for the Steiner problem in graphs, *Math. Japonica* Vol.24 No.6 (1980) pp. 573-577.

[32] S. Voss, Modern heuristic search methods for the Steiner tree problem in graphs, in Ding-Zhu Du, J.M. Smith, and J.H. Rubinstein (eds.) *Advances in Steiner Trees* (Dordrecht, Kluwer Academic Publishers, 2000) pp. 283-323.

[33] B.M. Waxman, Routing of multipoint connections, *IEEE Journal on Selected Areas of Communications* Vol.6 No.9 (1988) pp. 1617-1622.

[34] Q. Zhu, M. Parsa, and J. Garcia-Luna-Aceves, A source-based algorithm for delay-constrained minimum-cost multicasting, *Proceedings of IEEE INFOCOM'95* (Boston, MA, April 1995) pp. 377-385.