Baziakos, A.A.&. Economides, A.A.:**Multicast routing algorithms: a survey**. *Proceedings ICT '98, International Conference on Telecommunications*, pp. 476-480, 1998.

Multicast Routing Algorithms: A Survey

A. M. Baziakos and A. A. Economides

University of Macedonia, Greece
Department of Informatics

Abstract In this paper we survey multicast routing algorithms. First, we describe the multicast routing problem and the difficulties in routing multicast connections. Then, we provide a classification of multicast routing algorithms according to many different criteria. Finally, we present the main steps of several multicast routing algorithms, as well as comparison results among them.

Keywords: point-to-multipoint connection; multicast routing algorithm; Minimum Spanning Tree; minimum cost tree; shortest path tree; Steiner tree;

I. Introduction

In point-to-point networks, when a new connection is requested, a minimum "cost" (shortest) path connecting the source (sender) to the destination node is selected. Of course, the candidate paths should have sufficient unused bandwidth and should be able to provide the QoS (Quality of Service) required by the new connection. In networks with multipoint communication, the routing problem becomes even more complicated.

By multipoint communication, we refer to any type of communication within a group, that is, we refer to connections that involve more than two members. Namely, these are: i) point-to-multipoint connections, where there is one source (sender) and several destinations, as in broadcasting, ii) multipoint-to-point connections, where information from several sources is collected by one destination, as in data collection and iii) multipoint-to-multipoint connections, where each group member node can send and receive information to or from all other members, as in conference connections.

Since every multipoint-to-multipoint connection can be realized by a number of point-to-multipoint connections, we deal only with point-to-multipoint connections, or multicast connections, as we call them in this article. Therefore, the multipoint routing problem may be viewed as the decisions of (simultaneously) selecting the routes from the source node (sender) to the multiple destination nodes of the multicast group.

There are several approaches that can realize multicasting. The simplest approach but also the most expensive in terms of resources, is flooding, i.e. broadcasting packets to every node in the network. Another approach is to transmit a separate copy of the packet to each destination, which is still expensive. In an attempt to minimize the packet copies, we can either send multi-destination packets each including in its header multiple addresses, or partite the group members (nodes that belong to the multicast connection) according to locality and send one copy of the packet to each partition. Finally, another approach uses a tree that spans all group members, so instead of choosing the shortest path as in the point-to-point routing problem, we would choose the shortest tree that spans these members.

Among these approaches, the prevailing one is the use of trees, for two reasons: i) the packets can be transmitted in parallel to the various destinations along the branches of the tree and ii) a minimum number of copies of the packets are transmitted, with duplication of packets being necessary only at points where the tree diverges.

In the tree-based approach for multicasting, it is necessary that the switches have internal multicast capability. If the switch itself cannot generate multiple copies of the same packet, multicast connections can be established by a set of point-to-point connections, or by a central server. This server resides at some node (switch) and is responsible for duplicating the received packets and multicasting them to other group members. This technique can be extended to a network of multicast servers.

Another issue to be considered is the case of large number of groups and trees. In networks with only a few nodes, it is feasible to calculate in advance all possible trees for all possible user groups and store them in tables so that we can look them up when needed. However, for larger networks, this becomes inefficient. Rather, the tree has to be constructed on the fly for each multicast connection.

The remaining of this paper is organized as follows. In section II we present the objectives that multicast routing algorithms have to fulfill. Then, in section III we give a detailed taxonomy of routing algorithms. In section IV we give a short description of several algorithms. Next, in section V we give a comparison of the aforementioned algorithms and finally, in section VI we conclude.

II. Challenges in multipoint communication

In future high-speed networks, the users should have access not only to services for point-to-point applications, but also to new services for multipoint communication applications. Although the introduction of multipoint communication in commercial networks has been slow, it is now considered a very important mechanism in networks. The reason is that now there is a growing need for services for which multipoint communication is important, for example:

- News distribution (news agencies like the CNN distribute news reports from their headquarters to newspapers and radio stations throughout the world).
- Distribution of entertainment video.
- Multi-person conferences.
- Video lectures (tele-classrooms).
- Local Area Network interconnection (multipoint connection that makes several geographically distributed LANs to appear like one large network).
- Database update.
- Distributed games.
- Distributed Interactive Simulation.

Efficient multipoint routing, means first of all that as many multicast connection requests as possible must be routed (an algorithm that routes almost every connection request is superior to one that routes only some of the requests). Moreover, routing has to cope with dynamic (time varying) network topology and group membership. For example, in teleconferencing systems participants can join or leave a session when they wish to. Coping with such changes has to be done in a way that reliable transmission is provided (route changes or failure of network elements should have no side effects on the part of the tree that is used by other group members). Besides this, route optimality must be maintained after changes.

Other objectives to be met include: provision for the QoS constraints, minimization of the required buffer space in switches (otherwise, delivery to large groups is not feasible), minimization of the difference in delays perceived by any pair of receivers, low control message processing, as well as low packet processing. Finally, network resources must be used efficiently, while at the same time loops and traffic concentration on a link or a subnetwork must be avoided.

III. Classification of multicast routing algorithms

Multicast routing algorithms can be classified according to many different criteria. In this section we distinguish to:

- *algorithms that construct trees based on path costs* and *algorithms that construct trees based on link costs*. The link cost function can be the length of the link, the total delay encountered when traversing it (propagation, transmission and queuing delay), the monetary cost to the user of using capacity on the link, a function of the available capacity on the link, a function of the available buffer space, etc. A point to be noted here, is that minimizing the cost of a tree does not mean that the cost between the source and every destination is also minimum.
- *non-rearrangeable* and *rearrangeable algorithms*. In non-rearrangeable algorithms, once a particular set of edges has been used in a route, no reconfiguration is allowed. This means that adding a new member to the multicast group cannot result in any link being removed from the multicast connection and removing a member cannot result in any new links being added to a multicast connection. In rearrangeable algorithms, reconfiguration is allowed.
- *algorithms that support sparse node distribution* and *algorithms that support dense node distribution*. A group is said to be dense if there exist many group members within a region of an internet and sparse if there exist only a few group members within this region.
- *algorithms that are suitable for undirected networks* and *algorithms that fit well for directed networks*. In directed networks, the cost of the link connecting a node a to a node b is different than the cost of the link connecting node b to node a. In undirected networks, these costs are equal.
- *algorithms that implement source-based protocols* where a separate tree per sender in the group is generated and *algorithms that implement center-based protocols* where multiple senders to the group share the same tree. The latter approach is the most recent one.
- *distributed* and *centralized algorithms*. In distributed algorithms, each node acquires information only from neighboring nodes and makes a local routing decision. In centralized algorithms, a central node gathers all necessary information and makes a routing decision for the whole multicast group.
- *algorithms that solve the static problem* (group membership does not change over time, therefore new nodes cannot join or leave an existing multicast connection) and *algorithms that solve the dynamic problem*, where a sequence of requests to add or remove nodes from the multicast connection can be processed.
- *constrained algorithms*, that take into account QoS constraints such as delay or inter-destination delay variation constraints and *unconstrained algorithms*, that do not consider such constraints.
- *algorithms for bus-based networks* ([20] [21]) and *algorithms for point-to-point networks*. Networks can be either point-to-point networks

(using two-ended communication links), or can be composed of multiple-access media or buses (e.g. multichannel LANs, bus-based interconnection networks for parallel processors).

- *minimum cost algorithms* and *shortest path algorithms*. Minimum cost algorithms (like the numerous Steiner heuristics) try to minimize the cost of the multicast tree, which is the sum of the costs on the edges in the tree. The shortest path algorithms (like Dijkstra's or Bellman-Ford's algorithm) try to minimize the cost of every path from the source to a member of the multicast group.

IV. Short description of multicast routing algorithms

In this section, we briefly describe proposed multicast routing algorithms for point-to-point networks. It is assumed that point-to-point networks are presented as graphs $G=(V, E)$, where nodes (V) represent switching systems, edges (E) represent links and the edge lengths represent the costs associated with using the particular link. The set of destination nodes is named D and the source is named s. This presentation classifies the described algorithms according to whether they take into account QoS constraints or not.

A. Unconstrained algorithms

We start the description with these algorithms that do not take under consideration constraints. These algorithms consist the majority of the proposed algorithms and are the following:

Dijkstra's algorithm or Least-Cost (LC) algorithm [4]: It constructs a tree starting from the source node. At each step, it adds the nearest node to the source that is not already in the tree. The resulting tree spans all the nodes in the network. Dijkstra's algorithm is a shortest path algorithm.

Least Delay (LD) algorithm [12]: It uses Dijkstra's algorithm, assuming each link's delay to be the cost associated with that link. Therefore, it generates the tree that minimizes the delay of the path from the source to each group member separately.

Minimum Spanning tree (MST) [3] [5] [6]: First, it constructs another graph called complete graph containing just the nodes to be connected. All pairs of nodes in the complete graph are connected by an edge, with cost equal to the length of the shortest path in the original graph that connects this pair. Then it finds a tree of minimum cost that spans all the nodes of the complete graph and finally maps this tree onto the original graph. The MST algorithm is a centralized minimum cost Steiner heuristic for undirected networks.

Greedy algorithm [5]: It adds a new member to a tree by using the shortest path from that member to some node that is already part of the tree. When a member wishes to be removed from the multicast group, the portion of the tree that serves only this member is dropped. The Greedy algorithm is a distributed algorithm that is suited for dynamic multicast groups.

Weighted Greedy Algorithm (WGA) [6] [10]: It adds a new node u to a multicast connection by finding a node v that minimizes a combination of the distance from the new node u to the node v and the distance from the node v to the source s. So, it minimizes the function $W(v)=(1-\omega)\ d(u,v)+\omega\ d(v,s)$ where $0 \leq \omega \leq 0.5$, s denotes the source and $d(x,y)$ is the distance from x to y. The WGA algorithm is a distributed algorithm that is suited for dynamic multicast groups.

Rayward-Smith (RS) [16] [6] [9]: It detects potential Steiner points (non-destination nodes that are part of the Steiner tree solution and also have a degree in the Steiner tree that is equal or greater than three), which together with the set of destination nodes D, will provide the solution to the Steiner problem. The worst case bound for RS is two times the cost of an optimal solution [9]. The RS algorithm is a minimum cost Steiner heuristic .

Improved RS [6]: It runs RS a second time on the set of nodes generated by the initial RS solution.

Kou-Markowsky-Berman (KMB) [1] [6] [9] [19]: It uses the subgraph generated by the basic MST. This subgraph is created after mapping the paths that compose the complete graph's tree of minimum cost onto the original graph. This mapping provides a subgraph, which is the subgraph mentioned before). Then, it applies a minimum spanning tree algorithm to that subgraph. In the resulting tree, the branches that do not contain any nodes in D are pruned. Its solution has a cost that is never more than twice the optimal cost [9]. The KMB algorithm is a minimum cost Steiner heuristic for undirected networks.

Geographic Spread Dynamic Multicast (GSDM) [8]: It is based on spreading out the multicast connections on the basis of the link costs. The metric used to measure this spreading out is called Geographic Spread (GS). GSDM uses an intelligent shortest path algorithm, which incorporates GS and local re-routing if necessary. Emphasis is given in local rather than global optimization when a destination node is added or removed to/from the multicast group. The idea that motivated the implementation of GSDM was the effort to lower the cost of the multicast tree by means of spreading the multicast connections out "geographically" in the network. The greater the GS in the multicast connection, the more chances there are that a node wishing to join is close to a node already in the multicast connection, thus, improving local optimization, which in turn reduces the cost of the multicast tree. The GSDM algorithm is a rearrangeable algorithm that is suited for dynamic multicast groups.

Restricted Broadcast (RB) [11]: It constructs a minimal cost broadcast spanning tree (using the algorithm in [22]) and then prunes all the unnecessary edges. When a new node wishes to join the multicast connection, RB finds a minimum cost path that will connect the new node to the existing tree using only those links that belong to the broadcast spanning tree. The RB algorithm is suited for dynamic multicast groups.

Nearest Insertion (NI) [11]: It divides the network into subnetworks and designates a coordinator node for each subnetwork. The coordinator nodes are organized in a spanning tree (produced by the RB algorithm), which is used to send messages across subnetworks. When a node wants to join a group, it sends a request to its coordinator, which is also the root of the multicast subtree in the subnetwork. The coordinator asks all nodes in the subtree to report the length of their shortest path to the new node. The node that has the shortest path is chosen and the new member of the group is connected to it. The NI algorithm is suited for dynamic multicast groups.

Reverse Path Multicasting (RPM) [12]: It uses the shortest paths from each destination to the source to construct the multicast tree. The RPM algorithm is a distributed algorithm for dynamic multicast groups.
RNN/KMB, RNN/RS (RNN=Random Neural Network) [9]: It starts with the solution of KMB or RS and improves upon it, by adding vertices. The next vertex to be added is decided with the aid of a neural network. The heuristic tries to pick nodes that have low cost outcoming edges and are connected to as many nodes of the tree as possible. The RNN/KMB, RNN/RS algorithms are based on minimum cost Steiner heuristics.

Naive algorithm [1]: It finds the set of shortest paths between the source and destination nodes. If a link is used by more than one paths, then only one copy of the packets should be carried on that link, thereby reducing the total tree cost. The Naive algorithm is a minimum cost algorithm for directed networks.

Takahashi And Matsuyama (TAK) [1] [14]: First, it defines an initial subtree which consists of a single node (e.g. the source). Then the group member node with the lowest cost path to the existing subtree is added to the subtree. This process of adding one destination node at a time is repeated until there are not any more destination nodes. It is proven that the ratio of the worst case solutions to the optimal ones is no more than 2 [1]. The TAK algorithm is a minimum cost algorithm for directed networks.

Shortest Path Tree (SPT_{min} / SPT_{max}) [1]: First, it defines an initial subtree. This subtree consists of the highest cost path among the set of shortest paths that connect the source to the destination nodes. Then, from this set of shortest paths, the destination node with the lowest/ highest (SPT_{min} / SPT_{max}) cost path is added to the subtree. This process of adding one destination node at a time is repeated until there are not any more destination nodes. The SPT algorithm is a minimum cost algorithm for directed networks.

Dziong-Jia-Mason (DJM) [1]: First, it constructs a graph called complete graph containing only the nodes to be connected. All pairs of nodes in the complete graph are connected by an edge, with cost equal to the length of the shortest path in the original graph, connecting that pair. Then, it finds a tree of minimum cost called MST_1 , that spans all the nodes of the complete graph. In the next step, it defines an initial subtree which consists of the maximum cost edge in MST_1.
From that point on, the following process is repeated until no nodes remain to join the tree: First, it considers the edges of MST_1 whose one of their end-nodes belong already in the subtree. Among them, it selects the edge with the lowest value. Let the node at the other end of this edge be named v (this node can be the source or a destination node). Then, this edge is added to the subtree, provided that there is no transit node (not source or destination node) in the subtree that has a path to v with cost lower than the cost of this edge. Otherwise, the minimum cost path from this transit node to v is added. The ratio of the worst case solutions to the optimal ones is no more than 2 [1]. The DJM algorithm is a minimum cost Steiner heuristic for undirected networks.

B. Constrained algorithms

Next, we continue with these algorithms that **do** take under consideration constraints, such as the source-destination delay constraint (constraint on delay from the source to any destination node), or interdestination delay variation constraint

(constraint on the difference of any two source-destination delays). These algorithms are:

Kompella-Pasquale-Polyzos (KPP) [7]: First, it constructs a complete graph. Then, it generates a constrained spanning tree. In the final step, it expands the edges of the constrained spanning tree into the constrained cheapest paths they represent and removes any loops that may be caused by this expansion. It uses two functions in order to select the next node to be connected to the current subtree. The first function, f_{CD}, which corresponds to KPP_{CD}, explicitly uses both edge cost and delay, in an effort to choose low-cost edges that also have minimum delay. The second function, f_C, which corresponds to KPP_C, tries to construct the cheapest tree possible, while ensuring that the delay bound is met. The KPP algorithm is a Steiner heuristic for undirected networks.

Delay Variation Multicast Algorithm (DVMA) [13]: It produces constrained multicast trees. The constraints are: i) the source-destination delay constraint, ii) the interdestination delay variation constraint. First, it constructs the tree T_0 of shortest paths from s to all destination nodes using Dijkstra's algorithm. If T_0 does not satisfy the delay constraint, negotiation with destinations may be necessary to determine a looser delay constraint and construct a new tree. In case the delay constraint is met and the second constraint is also satisfied, a feasible tree has been found. Otherwise (when the second constraint is not met), DVMA is executed. DVMA operates by adding one destination node at a time to the existing subtree using a greedy approach. Among those paths that connect a new destination node u to the subtree and also satisfy the delay constraint, DVMA chooses the one that generates the tree that has the least value of interdestination delay variation. The DVMA algorithm is suited for static multicast groups.
A strategy that responds to dynamic changes of the multicast group is also presented in [13].

Constrained Bellman-Ford [12]: It uses a breadth-first search to find the constrained least-cost paths from the source to all other nodes in the network.

Constrained Adaptive Ordering (CAO) [12] [24]: It connects one group member at a time using the constrained Bellman-Ford algorithm. After each run of the constrained Bellman-Ford algorithm, the member with the minimum cost constrained path to the source that is not connected, is chosen. Then, it is added to the existing subtree. The links that compose this subtree are assigned a cost equal to zero. The CAO algorithm is a Steiner heuristic.

Bounded Shortest Multicast Algorithm (BSMA) [23] [12]: It computes a tree that minimizes the delay of the path from the source to each group member individually. Then it replaces superedges (paths between two branching nodes, or two multicast group members, or a branching node and a multicast group member) with superedges that have lower cost and are not part of the tree. The process of replacing superedges is repeated until the total tree cost cannot be reduced any more and it is done in such a way that the delay bound is not exceeded. BSMA uses a kth-shortest path algorithm to find lower cost superedges. The BSMA algorithm is a Steiner heuristic.

Constrained DijKStra heuristic (CDKS) [12] [25]: It computes a tree that minimizes the cost of the path from the source to each group member individually (least-cost tree). If the end-to-end delay to any group member does not meet the delay constraint, the respective path is replaced with the least-delay path. The least-cost tree is generated using Dijkstra's algorithm. The CDKS is a Dijkstra heuristic.

SemiConstrained heuristic (SC) [26] [12]: It uses the maximum end-to-end delay from the source to any node in the network as the delay constraint. It then constructs a broadcast tree that does not violate this constraint and prunes unnecessary parts. The SC algorithm is a Steiner heuristic.

Modified SemiConstrained heuristic (MSC) [12]: It is a version of SC that is closer to a semiconstrained shortest path tree. It has been found to perform better than SC with regard to tree costs, end-to-end delays and network balancing [12]. The MSC algorithm is a Steiner heuristic.

V. Comparison of multicast routing algorithms

In this section, we present performance evaluation comparison of the previously mentioned algorithms. The following results came from simulation experiments only, as it is practically not possible to attempt an analytical approach to the general problem of multicast routing. Furthermore, in many cases where simulation of the optimal algorithms was intractable due to excessive running times, the KMB heuristic was used in their place. The reason that motivated this choice is because its average case performance is close to optimal. In [6] for example, KMB showed near optimal average case performance for the models tested, exhibiting however worst case results up to 20% worse than optimal.

One case where KMB was used for comparison, is the one of [8], where it was found that GSDM exhibits a cost just slightly worse than that of KMB. As the size of the graphs increased, the mean cost also increased. Furthermore, for small multicast groups, GSDM did not perform well (almost two times worse than KMB). However, as the multicast group size approached the size of the graph, performance was improved, reaching that of KMB. GSDM was found to be much faster than KMB, even when more expensive paths were being considered by varying the GS threshold. GSDM did lower, albeit slightly, the number of packet copies per node as the Geographic Spread (GS) threshold was increased, at a cost of slightly decreasing performance.

Dziong in [1] compares KMB with TAK, SPT_{min}, SPT_{max}, DJM and the Naive heuristic. In his work, he takes in mind not only the tree cost, but also the transit nodes that the multicast tree entails. KMB and SPT_{min} give comparable cost results to TAK and DJM who have the smallest cost, while SPT_{max} is worse than them and the Naive heuristic is the worst among them. However, the Naive heuristic is good at constructing the shortest paths in terms of transit nodes in the tree. Although SPT_{max} produces trees with a lot more transit nodes, it still does better than SPT_{min} and TAK, which is almost two times worse than the Naive heuristic. Dziong suggests the use of SPT_{min}, as his results prove it to be a compromise between performance and complexity. Finally, he remarks that while the KMB heuristic exhibits increasing cost with increasing group size, the DJM heuristic, under the same circumstances, converges to the optimal solution.

Experimental results have shown that in terms of tree cost, the Greedy algorithm performs reasonably well for most cases when set against KMB [6], even though it is supposed to produce solutions that are far from optimal [5]. However, its tree cost increases after several nodes have ceased being in the multicast group.

On the other hand, the Weighted Greedy Algorithm (WGA), has better worst case cost performance than the Greedy algorithm and performs reasonably well for most cases in comparison to MST [6]. Still, it suffers from decreasing performance after several nodes have left the group, like the Greedy algorithm. As to the value of ω, it was found to have unimportant effect on performance [10], with ω in the neighborhood of 0.3 giving the best overall performance [6].

In [10], WGA with ω=0.5, which we call WGA(0.5) and is equivalent to shortest path routing to the source, is compared to a variation of MST called IMST. For light to moderate loads, WGA(0.5) produces blocking rates that can be as much as ten times the blocking rates of IMST for the same loads. However, for high loads, both algorithms have similar performance. Finally, it is shown that expensive requests (those that require large bandwidth and/or expensive paths) are blocked more often than inexpensive ones.

KMB, being an unconstrained algorithm, is most likely not to be able to fulfill the QoS requirements posed by real-time applications. Salama *et al* in [12], investigate how well unconstrained algorithms like KMB, RPM, Dijkstra's algorithm (LC) and the Least Delay (LD) algorithm can manage in real-time environments. In the same work the MSC (semiconstrained) and the KPP, CAO, BSMA and CDKS (constrained) heuristics are also examined under similar situations.

The conclusions drawn are as follows: KMB produces very low cost trees in the case of symmetric networks when compared to the optimal solution. However, in asymmetric networks, its performance is not so good. Still, it is always better than Dijkstra's algorithm (LC), as the latter does not try to optimize the total tree cost. The worst in all cases is the LD algorithm .

With regard to the delay constraint, KMB and the optimal minimum cost tree algorithm perform inadequately, whereas LC gives in some cases satisfactory results. This can be explained by the fact that finding the least cost path to each destination node results in minimizing the number of hops traversed by this path, which indirectly reduces delay along that path. LD provides solutions that meet the delay bound.
KMB can route more connection requests than LC and LD.

RPM is slightly worse than LC in terms of cost when the network asymmetry is small and about 30% worse but as the than KMB, asymmetry increases it becomes as much as 80% worse than KMB. RPM has similar delay performance with LC. If resource reservation and admission control is combined with RPM routing, its ability to route connection requests is dramatically improved.

The constrained algorithms KPP, CAO and BSMA construct trees that meet the delay bound (with CDKS giving the best results) and moreover they produce low cost trees (with BSMA generating the lowest cost trees among them). In addition, they are efficient in routing multicast connections. BSMA is better than CAO and KPP with regard to the number of multicast connections it can route.

The constrained Dijkstra heuristic, CDKS, does not perform as well as the previous three heuristics both in cost issues and network resources issues, but its tree costs are always within 25% of optimal.

The semiconstraint algorithm, MSC, does not construct low cost trees and is not as efficient in routing multicast connection requests as the other three constraint Steiner heuristics. However, it meets the delay constraint (MSC's maximum end-to-end delays are comparable to LD's maximum end-to-end delays).

All three constraint Steiner heuristics have high execution times. As the network size increases, BSMA's average execution time grows much faster than KPP's and CAO's. The latter has fast execution times for small group sizes. As to the worst-case scenario, KPP and CAO exhibit execution times that grow exponentially with the network size, while BSMA shows polynomial growth in the network size. On the other hand, LD, LC and RPM have fast execution times, which also holds for the well-scaling-to-large-network-sizes CDKS and MSC heuristics. It seems that CDKS achieves a good compromise between reasonable tree costs and fast execution times. Therefore, it is suggested in [12] as a possible solution for the multicast routing problem.

Another constraint algorithm, DVMA [13] is compared with three algorithms, namely: i) Dijkstra's Least-Cost algorithm, ii) Prim's algorithm (which constructs a tree of minimum weight spanning all nodes in the network, that is then pruned) and iii) the TraDeofF (TDF) algorithm [18] between the minimum spanning tree heuristic for the Steiner tree problem [17] and LC.

DVMA constructs trees that have a maximum delay variation that is always smaller than that of the LC, TDF and Prim trees. The Prim algorithm produces by far the worst tree, while TDF and LC are somewhere in the middle. As the size of the multicast group increases, the improvement of DVMA over LC decreases. DVMA is also affected by the average nodal degree, showing dramatic improvement as the nodal degree increases, while the rest algorithms are not significantly affected by it. The conclusion drawn here is that DVMA is suited for small size multicast groups and /or high nodal degrees.

For dynamic groups, the dynamic version of DVMA performs better than LC and worse than running DVMA anew each time the group changes. This suggests that, if the number or join or leave requests to/from the multicast group is not very large, the dynamic approach performs reasonably well.

Even though DVMA has good average case behavior in terms of maximum delay variation, the cost of the produced tree may be high, as the algorithm does not attempt to minimize the tree cost.

The KPP, is a heuristic that always finds a constrained spanning tree if one exists. KPP's two versions [7], may find paths with delays far lower than the maximum allowed delay, at the expense of added cost to the tree. KPP_C has better average tree cost performance than KPP_{CD} . However, both versions converge to the minimum spanning tree as the multicast group increases in size. Additionally, the tree cost of both of them converges as the delay tolerance increases. LD, the shortest delay tree algorithm, produces trees with consistently high costs approaching the double of KPP_C's cost.

The RS heuristic, gave near optimal average case results for the two graph models considered in [6]. In [1] it was also stated that statistically the RS heuristic performs better than KMB. In [9], RS was found once again to be better than KMB. E. Gelenbe *et al* in [9] proposed the use of Random Neural Networks (RNNs) to improve upon the solutions of KMB and RS. Indeed, both RNN/RS and RNN/KMB gave better results than RS and KMB respectively. In fact, RNN/RS found the best solution for the majority of the graphs tested. All algorithms examined in [9] became progressively better as the group size approached the size of the network. In [6], an improved version of RS gave performance very close to optimal for the entire set of experiments conducted.

The last two of the algorithms presented in the previous section are the RB and the NI algorithms [11]. RB has higher tree cost than NI. When compared to the algorithm proposed by Wall in [15] which generates trees that are close to the optimal ones, both of them have greater tree cost (around 10% for large groups and 20% for small and medium groups). As to the total cost (the one that includes the cost of control messages that are needed to update the tree), Wall's algorithm generates trees of rapidly increasing cost as the group size increases and/or the behavior of the group becomes more dynamic. On the contrary, both RB and NI have similar cost and are relatively independent of the degree of dynamic behavior for all group sizes. Moreover, they scale progressively in cost with increasing group size and have low update delay.

VI. Conclusions

Multicast routing is definitely a problem that seeks further investigation. Numerous algorithms and heuristics can be found in the literature, each being suitable for a particular case and none suitable for every case. However, as this topic is attracting even more attention, we can hope for better solutions to come, in the sense that they can be better suited for the general case.

At this point in time, there exist heuristics that give results close to optimal for a particular case. These heuristics can be used in place of the corresponding optimal algorithms as a reference to compare with, since the optimal algorithms exhibit execution times that become far to large when the network size increases.

In general, even though some algorithms may provide good results, they may not be the best choice to implement in a real network, because of their slow execution. Unfortunately, we must "sacrifice" a part of the efficiency for the benefit of speed.

Since there is a strong demand for real-time applications, constrained algorithms become crucial for efficient multicast routing. Furthermore, integration of routing with resource reservation and admission control should also be investigated. We believe that all the above issues and problems should be treated in an integrated way and not each one separately from the other.

References

[1] Z. Dziong, *ATM Network Resource Management*, The McGraw-Hill Series on Computer Communications, 1997.

[2] 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, pp. 277-290, April 1997.

[3] A. G. Waters, "Multicast Provision for High Speed Networks", *High Performance Networking*, IV (C-14), A. Danthine and O. Spaniol (Editors), Elsevier Science Publishers B. V. (North-Holland), pp. 317-332, IFIP 1993.

[4] A. Gibbons, *Algorithmic Graph Theory*, Cambridge University Press, 1985.

[5] J. S. Turner, "The Challenge of Multipoint Communication", *Traffic Engineering for ISDN Design and Planning*, M. Bonatti and M. Decina (Editors), Elsevier Science Publishers B. V. (North-Holland), pp. 263-279, IAC 1988.

[6] B. M. Waxman, "Routing of Multipoint Connections", *IEEE Journal on Selected Areas in Communications*, Vol. 6, No. 9, pp. 1617-1622, Dec. 1988.

[7] V. P. Kompella, J. C. Pasquale and G. C. Polyzos, "Multicast Routing for Multimedia Communication", *IEEE/ACM Transactions on Networking*, Vol. 1, No. 3, pp. 286-292, June 1993.

[8] J. Kadirire, "Minimizing Packet Copies in Multicast Routing by Exploiting Geographic Spread", *ACM SIGCOMM, Computer Communication Review*, Vol. 24, No. 3, pp. 47-62, July 1994.

[9] E. Gelenbe, A. Ghanwani and V. Srinivasan, "Improved Neural Heuristics for Multicast Routing", *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 2, pp. 147-155, Feb. 1997.

[10] B. M. Waxman, "Performance Evaluation of Multipoint Routing Algorithms", *Proc. IEEE INFOCOM*, pp. 980-986, 1993.

[11] N. E. Belkeir and M. Ahamad, "Low Cost Algorithms for Message Delivery in Dynamic Multicast Groups", *Proc. IEEE ICDCS*, pp. 110-117, 1989.

[12] 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 in Communications*, Vol. 15, No. 3, pp. 332-345, April 1997.

[13] G. N. Rouskas and I. Baldine, "Multicast Routing with End-To-End Delay and Delay Variation Constraints", *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 3, pp. 346-356, April 1997.

Further Reading

[14] H. Takahashi and A. Matsuyama, "An Approximate Solution for the Steiner Problem in Graphs", *Math. Japonica*, Vol. 24, No. 6, pp. 573-577, Feb. 1980.

[15] D. W. Wall, "Mechanisms for Broadcast and Selective Broadcast", Ph.D. dissertation, Stanford Univ., Dept. Elec. Eng., Tech. Report No. 190, June 1980.

[16] V. J. Rayward-Smith, "The Computation of Nearly Minimal Steiner Trees in Graphs", *Int. Journal Math. Educ. Sci. Technol.*, Vol. 14, No. 1, pp. 15-23, Jan./Feb. 1983.

[17] E. N. Gilbert and H. O. Pollak, "Steiner Minimal Trees", *SIAM J. Appl. Math.*, Vol. 16, No. 1, pp. 1-29, 1968.

[18] K. Bharath-Kumar and J. M. Jaffe, "Routing to Multiple Destinations in Computer Networks", *IEEE Trans. Commun.*, Vol. COM-31, No. 3, pp. 343-351, Mar. 1983.

[19] L. Kou, G. Markowsky and L. Berman, "A Fast Algorithm for Steiner Trees", *Acta Informatica*, Vol. 15, No. 2, pp. 141-145, 1981.

[20] P. K. McKinley and J. W. S. Liu, "Multicast Tree Construction in Bus-Based Networks", *Communications of the ACM*, Vol. 33, No. 1, Jan. 1990.

[21] P. K. McKinley, "Multicast Routing in Spanning Bus Hypercubes", *Proceedings of the 1988 Int. Conf. on Parallel Processing*, Aug. 1988.

[22] R. G. Gallager, P. A. Humblet and P. M. Spira, "A Distributed Algorithm for Minimum Weight Spanning Trees", *ACM Trans. on Programming Languages and Systems*, Vol. 5, No. 1, pp. 66-77, Jan. 1983.

[23] Q. Zhu, M. Parsa and J. J. Garcia-Luna-Aceves, "A Source-Based Algorithm for Near-Optimum Delay-Constrained Multicasting", *Proc. IEEE INFOCOM '95*, pp. 377-385, Mar. 1995.

[24] R. Widyono, "The Design and Evaluation of Routing Algorithms for Real-Time Channels", Int. Comp. Sci. Institute, Univ. of California at Berkeley, Tech. Rep. ICSI TR-94-024, June 1994.

[25] Q. Sun and H. Langendoerfer, "Efficient Multicast Routing for Delay-Sensitive Applications", *Proc. Second Workshop Protocols Multimedia Systems (PROMS '95)*, pp. 452-458, Oct. 1995.

[26] A. G. Waters, "A New Heuristic for ATM Multicast Routing", *Proc. Second IFIP Workshop Performance Modeling Evaluation ATM Networks*, pp. 8.1-8.9, July 1994.