Multicast Routing Algorithms and Protocols: A Tutorial

Laxman H. Sahasrabuddhe and Biswanath Mukherjee, University of California

Abstract

Multicasting is the ability of a communication network to accept a single message from an application and to deliver copies of the message to multiple recipients at different locations. Recently, there has been an explosion of research literature on multicast communication. This work presents a tutorial-cum-survey of the various multicast routing algorithms and their relationship with multicast routing protocols for packet-switched wide-area networks. Our contribution should be of particular benefit to the generic networking audience (and, to a lesser extent, to the expert on this subject).

Multicasting is the ability of a communication network to accept a single message from an application and to deliver copies of the message to multiple recipients at different locations. One of the challenges is to minimize the amount of network resources employed by multicasting. To illustrate this point, let us assume that a video server wants to transmit a movie to 1000 recipients (Fig. 1a). If the server were to employ 1000 separate point-to-point connections (e.g., TCP connections), 1000 copies of the movie may have to be sent over a single link, thus making poor use of the available bandwidth. An efficient implementation of multicasting permits much better use of the available bandwidth by transmitting at most one copy of the movie on each link in the network, as shown in Fig. 1b.

Recently, there has been a lot of research in the area of multicast communication. Although many excellent surveys and books exist which examine various aspects of multicasting [1–6], in the course of our studies we have found a need for a tutorial-cum-survey of the various multicast routing algorithms and their relationship with multicast routing protocols. In this work we present a tutorial-cum-survey of the following two important topics in multicasting:

- Multicast routing algorithms
- Multicast routing protocols

Communication networks can be classified into two categories: local area networks (LANs) and wide area networks (WANs). A LAN spans a small geographical area, typically a single building or a cluster of buildings, while a WAN spans a large geographical area (e.g., a nation). Often, nodes connected to a LAN communicate over a broadcast network, while nodes connected to a WAN communicate via a switched network. In a broadcast LAN, a transmission from any one node is received by all the nodes on the network; thus, multicasting is easily implement-

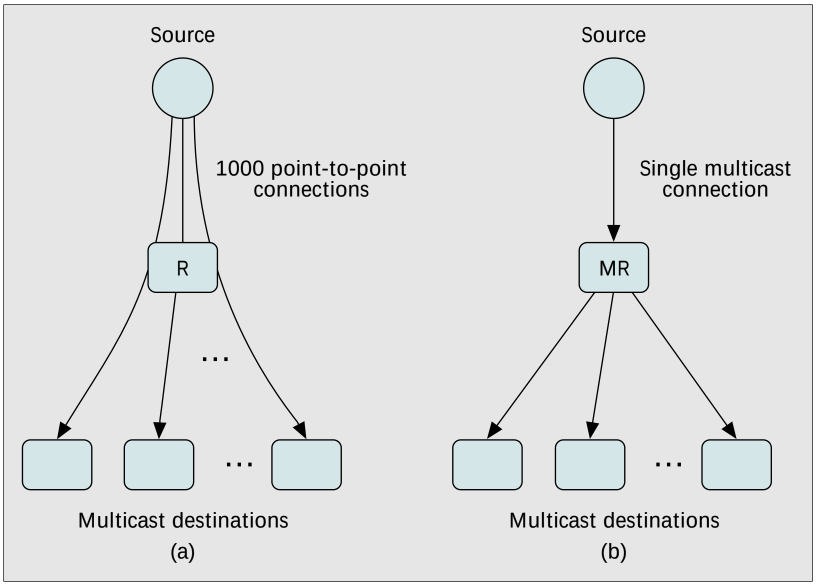
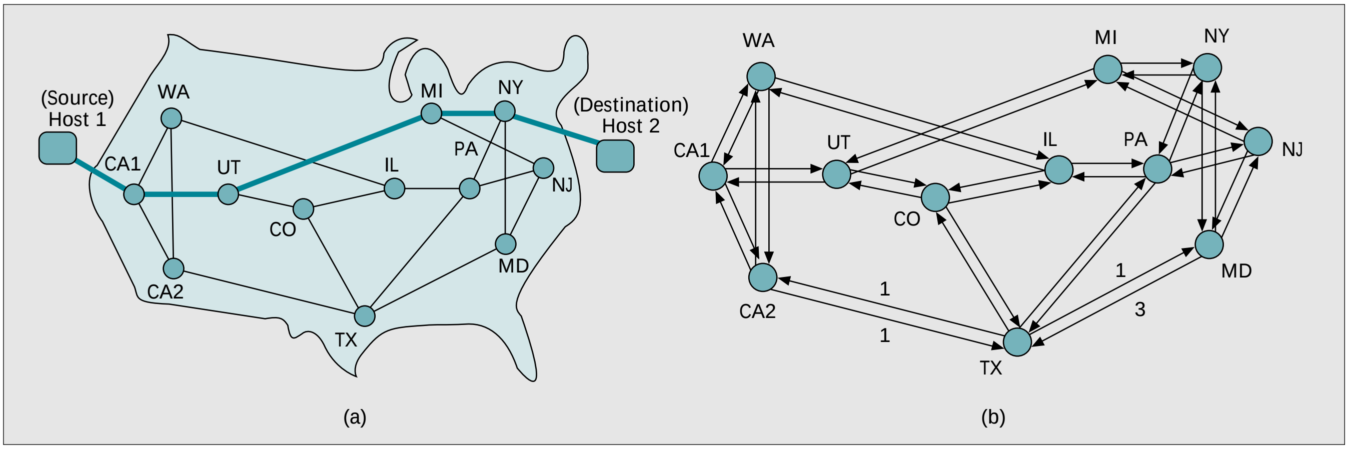
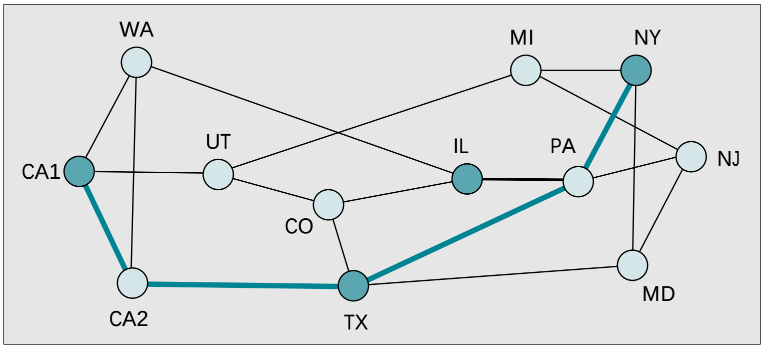


Figure 1. *An example illustrating the amount of network resources employed: a) unicasting a movie to 1000 different users; b) multicasting the movie.* (R = *standard router*, MR = *multicast router.)*

This work has been supported in part by the National Science Foundation (NSF) under Grants Nos. NCR-9508238 and ANI-9805285.



■ Figure 2. *a) An example of a WAN. Data is transmitted from source Host 1 to destination Host 2 through the route consisting of nodes* CA1, UT, MI, *and* NY; *b) a directed graph that models the WAN shown in a).*



■ Figure 3. *An example of a Steiner tree. Multicast group =* (CA1, TX, IL, NY)*. Cost of all links = 1. Cost of Steiner tree = 5.*

ed on a broadcast LAN. On the other hand, implementing multicasting on a switched network is quite challenging; hence, throughout this work, we will focus on the multicasting problem in a WAN which is based on a switched network.

Today, many multicast applications exist, such as news feeds, file distribution, interactive games, and videoconferencing, but the implementation of these applications is not necessarily efficient because today's WANs were designed to mainly support point-to-point (unicast) communication. In the future, as multicast applications become more popular and bandwidth-intensive, there will emerge a pressing need to provide efficient multicasting support on WANs.

A WAN consists of nodes (i.e., switches or routers) interconnected by communication links. A transmission from a source to a destination is routed through these interconnected nodes. Figure 2a shows an example of a route of a transmission from a source to a destination on a WAN.[1]

A WAN can be modeled by a *directed graph*. Figure 2b shows a directed graph that models the communication network shown in Fig. 2a. A directed graph consists of a set of nodes V and a set of links E. A link connecting node u to node v is represented by an ordered tuple (u, v). Nodes in the directed graph represent nodes in the WAN, while links in the directed graph represent communication links in the WAN. (Note that the graph in Fig. 2b is a special one in the sense that if there is a link (u, v), there also exists a link (v, u); this characteristic, however, is not a necessity in a general directed graph.)

Communication links in a network may have different properties. For example, a fiber optic communication link may have very large bandwidth compared to a copper wire communication link. A property of a communication link is represented by a *weight* of the corresponding link in a graph. For example, if the propagation delay of the communication link ($CA2$,TX) is 1 ms, this information can be represented by assigning a weight equal to 1 to the link ($CA2$,TX) in Fig. 2b, with the weights of the other links being their corresponding propagation delays in milliseconds.

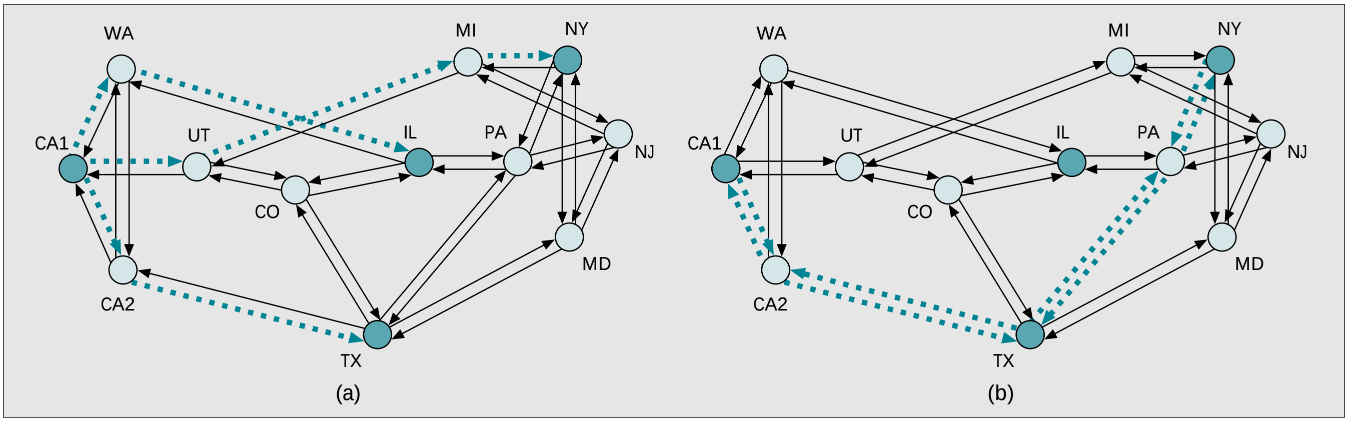
The communication links in Fig. 2 can be of two types: *symmetric* and *asymmetric*. Symmetric links have the same weight in both directions, while asymmetric links have different weights depending on the direction. Thus, in Fig. 2b, which shows weights on only four links, the link between nodes $CA2$ and TX is symmetric, while the link between nodes TX and MD is asymmetric. If all the links in a WAN are symmetric, we can model the WAN by an *undirected graph*, as shown in Fig. 3. In an undirected graph, the direction of a link is unimportant; hence, a link between node u and node v can be represented by an unordered tuple (u, v). Traditionally, communication networks have been modeled by undirected graphs. Henceforth in this work, unless otherwise stated, the term *graph* will refer to an undirected graph.

In unicast (point-to-point) communication, routing is often treated as the *shortest-path problem* in graphs. When two nodes wish to communicate, a *minimum-weight path* (*shortest path*) connecting the corresponding pair of nodes is selected. In multicasting, a group of more than two nodes (also called the multicast group) wish to communicate with one another. Now, instead of the shortest path, we are interested in the *minimum-weight tree* which spans all the nodes in the multicast group.

In general, different multicast applications have different requirements. For example, a reliable data transfer multicast application, such as software distribution, has very different requirements from a real-time multimedia multicast application, such as nationwide videoconferencing. Thus, it is helpful to classify multicast communication into two types:

- Source-specific: In source-specific multicast communication, only one node in the multicast group sends data, while all the other nodes receive data.
- Group-shared: In group-shared multicast communication, each node in the multicast group can send data to the mul-

[1] *In the remainder of this work, (a) we will use the terms "switch" and "router" interchangeably, and (b) the combination of a host and the corresponding switch through which it interfaces to the network will be treated as an integrated unit and will be referred to as a node. For example, (source) host Host 1 and switch CA1 will be referred to as (source) node CA1.*



■ Figure 4. *An example of a) a source-specific tree that employs unidirectional links, and b) a group-shared tree that employs bidirectional links.*

ticast group as well as receive data from other nodes in the multicast group.

The next section discusses multicast routing algorithms. We then study the implementation of multicast routing protocols on the Internet. Note that the current Internet uses IPv4, while the next-generation Internet (NGI) will employ IPv6. Since some topics discussed are specific to IPv4, they are not applicable to the NGI, although the general principles discussed will still be applicable. On the other hand, the subsections on multicast routing algorithms are relevant to both IPv4 and IPv6 because they do not presuppose any particular network-layer protocol. Finally, we provide concluding remarks.

Multicast Routing Algorithms

Figure 3 shows an undirected graph $G = (V, E)$, where V is the set of nodes and E the set of links. Note that, since graph G is undirected, it models a communication network which has symmetric links. Let $M = (CA1, TX, IL, NY)$ be a multicast group. (Shaded nodes in Fig. 3 belong to the multicast group.) Now, in order to perform multicast communication, the nodes in the multicast group must be interconnected by a tree. Thus, *the problem of multicast routing in communication networks is equivalent to finding a tree* T *in graph* G *such that* T *spans all vertices in the multicast group* M. Such a tree is called a *multicast tree* and is shown in Fig. 3 by thick lines.[2] (The term *Steiner tree* used in Fig. 3 will be clarified next.)

Just as multicast communication can be of two types, multicast trees can also be classified into two corresponding categories: source-specific (or source-rooted) and group-shared. For the same multicast example as in Fig. 3, Fig. 4a shows a source-specific multicast tree which employs unidirectional links[3] (with source = $CA1$), while Fig. 4b shows a group-shared multicast tree. The key difference between a source-specific multicast tree and a group-shared multicast tree is that a source-specific multicast tree is optimized for source-specific multicast communication, while a group-shared multicast tree is optimized for group-shared multicast communication. For example, if we want to minimize the average delay for source-specific communication, we need to minimize the average source-specific delay which is calculated by taking the average of the end-to-end delays over all (*source*, *multicast-member*) pairs. Now, assuming that each link in Fig. 4a has delay equal to 1, the source-specific delay of the source-specific tree rooted at $CA1$ is equal to 2.33 (the average of the delay from source $CA1$ to nodes TX, IL, and NY). In comparison, the source-specific delay (with $CA1$ as the source) of the group-shared multicast tree shown in Fig. 4b is equal to 3.33. On the other hand, if we were to calculate the average group-shared delay of the source-specific tree by taking the average of the end-to-end delays over all (*multicast-member, multicast-member*) pairs, the average group-shared delay is equal to 3.5 in Fig. 4a, while the average group-shared delay of the group-shared tree in Fig. 4b is equal to 2.67. Thus, the application requirements dictate which type of multicast trees are "better."

The following is a list of the properties of a good multicast tree. Since for most multicast applications some properties are more important than others, we have divided the properties into three priority levels:[4] high, medium, and low.

High Priority

- Low cost: The cost (or weight) of a multicast tree is the sum of the costs (or weights) of all the links in the multicast tree. A good multicast tree tries to minimize this cost.
- Low delay: The end-to-end delay from the source node to the destination node is the sum of the individual link delays along the route. A good multicast tree tries to minimize the end-to-end delay for every source-destination pair in the multicast group.
- Scalability: A good multicast tree is scalable in two respects. First, constructing a multicast tree for a large multicast group should require reasonable amounts of time and resources. Second, the switches in the communication network should be able to simultaneously support a large number of multicast trees.

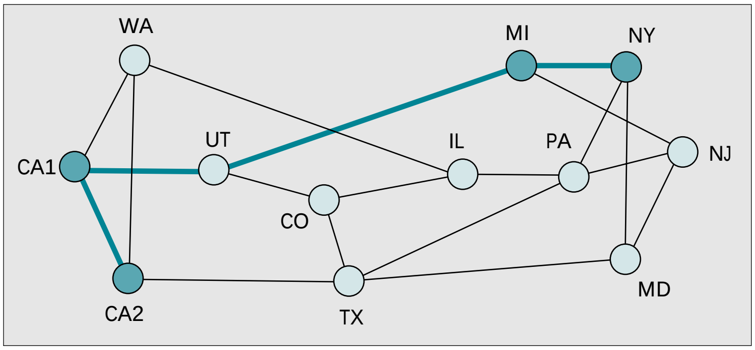
Medium Priority

- Support for dynamic multicast groups: Multicast groups can be classified as static and dynamic. The members of a static multicast group do not change over time; in a dynamic multicast group, new members may join or existing members leave. A good multicast tree should allow multicast members to join or leave the multicast tree in a seamless fashion. Moreover, the properties of a good multicast tree should not degrade due to the dynamic nature of the multicast group.
- Survivability: A good multicast tree should be able to survive multiple node and link failures.

[2] *Throughout this work, the default weight of all links, unless specified otherwise, is equal to 1.*

[3] *Note that a source-specific multicast tree connects a source node to other nodes in the multicast group by employing either unidirectional or bidirectional links, while a group-shared multicast tree employs only bidirectional links.*

[4] *Note that the priority levels may be different for certain applications. For example, while the fairness property of a multicast tree is not very important in general, it may be the most important property of the multicast tree if the multicast tree is being employed by a multiplayer game. Moreover, although some properties are considered low-priority for today's applications, they may become more important in the future due to emerging applications which may be beyond our comprehension today.*



■ Figure 5. *An example of a graph for which the Steiner tree can be found by employing the reduction rules. Cost of all links = 1; cost of the Steiner tree = 4.*

Low Priority

- Fairness: A good multicast tree is fair in two respects. First, it tries to provide a minimum quality of service (e.g., bounded delay) to each member in the multicast group. (It is not fair to unnecessarily punish one member in order to improve the quality of service to other members.) Second, it tries to evenly divide the multicasting effort (e.g., packet duplication effort) among the participating nodes.

Most algorithms that have been proposed in the literature mainly focus on cost and delay optimization, although the other properties have also been addressed to a lesser extent. Before we examine each of the above properties in detail, let us examine some important theoretical concepts and definitions which will help us better understand the nature of the multicast routing problem.

The classical optimization problem in multicast routing is called the *Steiner tree problem in networks* (*SPN*), and is defined as follows. Given

- An undirected graph $G = (V, E)$
- A cost function which assigns a positive real cost c_{uv} to link (u, v)
- A set of nodes $M \subseteq V$ which belong to the multicast group

find a tree $T = (V_T, E_T)$ which spans M, such that its cost $C_T = \Sigma_{(u, v)\in E_T} c_{uv}$ is minimized. Such a minimum-cost multicast tree is called a Steiner tree. Note that since graph G is undirected, it models a communication network which has bidirectional links; thus, Steiner tree T is a group-shared multicast tree.

Figure 3 shows a Steiner tree which connects the multicast group consisting of nodes *CA*1, *TX*, *IL*, and *NY*. If we assume the cost of each link to be equal to 1, the cost of the Steiner tree will be equal to 5. Note that nodes *CA*2 and *PA* do not belong to the multicast group, but are part of the Steiner tree. Such nodes are called *Steiner nodes.*

Although SPN is NP-complete [7], there are some trivial[5] cases of SPN that can be solved in polynomial time, as shown below [1]:

- $|M| = 2$ (unicast case): There are only two nodes in the multicast group. SPN reduces to the well-known shortest-path problem. Polynomial-time algorithms for this problem are known [8, 9].
- $|M| = |V|$ (broadcast case): In this case, the multicast group contains all the nodes in the network. Thus, SPN reduces to the well-known minimum spanning tree problem. Polynomial-time algorithms for this problem are known [10, 11].
- G is a tree: In this case, there is only one subtree which spans the multicast group M; this subtree is the solution to SPN.

Moreover, for certain cases of SPN, we can reduce the size of the problem by employing the following rules [1]. Note that each rule can be performed in polynomial time. Let deg(v) denote the degree of the node $v \in V$.

1)If G contains a node v with $\deg(v) = 1$, then v and the link (u, v) can be removed from G. If $v \in M$ and $u \notin M$, then u is added to the multicast group in the reduced graph. Note that, if $v \in M$, then link (u, v) belongs to the Steiner tree.

2)If G contains a node $v \notin M$ with $\deg(v) = 2$, then the two links (i, v) and (v, j) can be replaced by a link (i, j) of cost $c_{ij} = c_{iv} + c_{vj}$. If, as a result, two links become parallel, the one with the larger cost can be removed from G.

3)If G contains a link (i, j) such that $c_{ij} > d_{ij}$, where d_{ij} is the cost of the shortest-path between nodes i and j, then link (i, j) can be removed from G. Furthermore, if $c_{ij} = d_{ij}$ and there is a path of cost d_{ij} from i to j not containing (i, j), then link (i, j) can be removed from G.

4)If G contains three distinct nodes $u, v, w \in M$, such that u and v are adjacent, $c_{uv} > d_{wu}$, and $c_{uv} > d_{wv}$, then link (u, v) can be removed from G. In other words, if u, v, and w are any three nodes in the multicast group such that the cost of the link (u, v) is more than the cost of a path from node w to node u as well as node v, then link (u, v) does not belong to the Steiner tree.

5)Let $u \in M$. Let v and w be the closest and second closest adjacent nodes to u, respectively. Now, if $c_{uv} + \min\{d_{vp} | p \in M$ and $p \neq u\} \leq c_{uw}$, then the link (u, v) belongs to the Steiner tree and G can be contracted along (u, v). In other words, if the closest adjacent node (v) brings you nearer to other members of the multicast group, then link (u, v) should belong to the Steiner tree.

For example, the graph shown in Fig. 5 (nodes in the multicast set are shaded) can be reduced to a single node by employing reduction 5 repetitively as follows. First, we contract along link (*CA*2, *CA*1); second, we contract along link (*MI*, *NY*); third, we contract along link (*CA*1, *UT*); and finally, we contract along link (*UT*, *MI*).

Unfortunately, as the following lemma demonstrates, these reductions cannot be applied to a large number of instances of SPN which occur in typical communication networks. Usually, these reductions cannot be applied to cases in which $|M| \ll |V|$, G is not sparse,[6] and G satisfies the triangle inequality (to be explained shortly). The following Lemma describes a sufficient condition for an instance of SPN to be "irreducible."

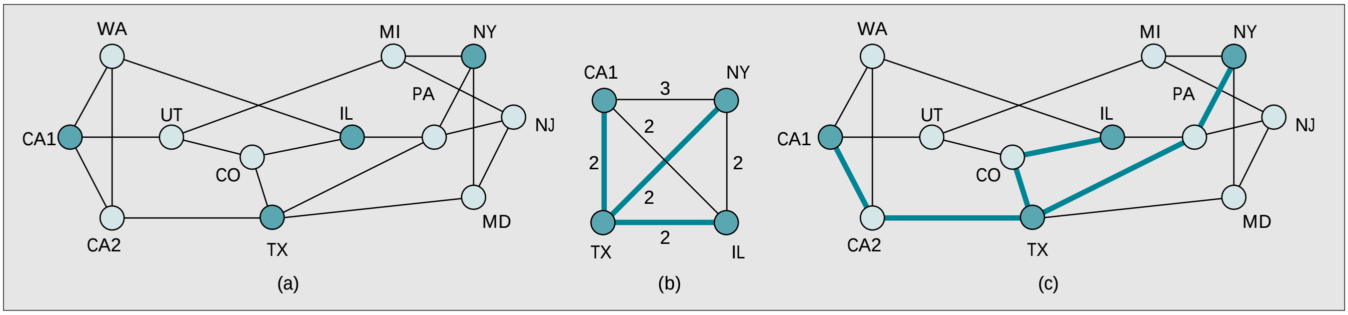
Lemma 1 — If an instance of SPN (say P) satisfies *all* of the following three conditions, then P cannot be reduced to a smaller instance of SPN by using the aforementioned reduction rules.

1. The graph satisfies the triangle inequality, that is, the cost c_{uv} of a link (u, v) is strictly less than the cost of any path from node u to node v which does not include link (u, v).
2. The minimum degree of the graph is 3, that is, $\forall v \in V$, $\deg(v) \geq 3$.
3. None of the nodes in the multicast group are adjacent to one another, that is, $\forall u, v \in M, (u, v) \notin E$.

Proof — Reductions 1 and 2 cannot be applied because the minimum degree of the graph is 3. Reduction 3 cannot be applied because the graph satisfies the triangle inequality. Reduction 4 cannot be applied because none of the nodes in the multicast group are adjacent to one another. Finally, reduction 5 cannot be applied because the graph satisfies the triangle

[5] *There are some other special cases of SPN for which polynomial-time algorithms exist [1].*

[6] *We define graph* G *to be sparse if all the spanning trees of graph* G *can be enumerated in polynomial time.*



■ Figure 6. *An example of a Steiner tree constructed by employing the KMB algorithm.*

inequality, and none of the nodes in the multicast group are adjacent to one another. The graph shown in Fig. 3 satisfies all the above conditions; thus, it is an “irreducible” graph. ■

Thus, for typical communication networks, it may be impossible to find a Steiner tree in a reasonable amount of time; hence, it is important to develop *approximation algorithms* for SPN.

Approximation algorithms for SPN run in polynomial time and produce good-quality (but not necessarily optimal) solutions to SPN. For some approximation algorithms, it is possible to prove a *performance guarantee* (i.e., a bound on the quality of the solution). A formal definition of performance guarantee is as follows. Let Γ be a class of problems (such as SPN) and $P \in \Gamma$ be a problem instance. Let $A(P)$ denote the cost of the solution found by algorithm A and $OPT(P)$ denote the cost of the optimal solution. We define the *performance guarantee* of algorithm A as $\Pi_A = \max_{P \in \Gamma}\{A(P)/OPT(P)\}$. In other words, if the performance guarantee of an algorithm is equal to β, then for all problem instances $P \in \Gamma$, the approximate solution is guaranteed to be at most β times costlier than the optimal solution. While most approximation algorithms for SPN have a performance guarantee of 2, to the best of our knowledge, none of the known approximation algorithms have a performance guarantee better than 11/6 [12]. In the following subsections, we examine the six properties of a multicast tree that were mentioned at the beginning of this section, paying more attention to the higher-priority properties, particularly cost and delay.

Cost Optimization

Approximation algorithms for optimizing the cost of a multicast tree employ different kinds of heuristics [13–16]. Recall that if the multicast group consists of all the nodes in the graph, the problem reduces to the well-known minimum spanning tree problem. Thus, it is no surprise that some approximation algorithms are based on the so-called minimum spanning tree heuristic. One such approximation algorithm which was proposed by Kou, Markowsky, and Berman (henceforth referred to as KMB) [13] is examined below.

KMB consists of five steps. First, using the nodes in the multicast group, we construct an undirected closure graph G_1; thus, for every node pair (u, v) in the multicast group M, G_1 has an edge (u, v), such that the weight of the edge (c'_{uv}) is equal to the weight of the shortest path (d_{uv}) between nodes u and v in G. Second, we find the minimum spanning tree of the closure graph G_1. Third, we construct graph G_2 by replacing each link in the spanning tree of G_1 by the corresponding shortest path in G. Next, we find the minimum spanning tree T_2 of graph G_2. Finally, we construct the multicast tree T_M by deleting links in T_2, if necessary, in such a way that all the leaves in T_M belong to the multicast group.

Figure 6a shows a graph G and the multicast group M (shaded nodes). Figure 6b shows the corresponding undirected closure graph with thick lines corresponding to the minimal spanning tree T_1 (after applying steps 1 and 2 from above). Finally, Fig. 6c shows the Steiner tree in thick lines (after applying steps 3, 4, and 5). The KMB algorithm has a performance guarantee of $2(1 - 1/|M|)$.

Recall that the KMB algorithm assumes that the communication network has symmetric link costs. Given the increasing heterogeneity of applications and communication links (e.g., satellite and radio links are becoming common), the link costs may be asymmetric; that is, the cost of a link between any two adjacent nodes is not the same in both directions. In a communication network with asymmetric links, the problem of finding a minimum-cost group-shared multicast tree can be reduced to SPN as follows. Let $G = (V, E)$ be a directed graph which models a communication network with asymmetric links. Now, construct an undirected graph $G' = (V, E')$ (note that G and G' have the same set of vertices) such that for every pair of directed links (u, v) and (v, u) in G, there is a corresponding undirected link (u, v) in G' which has a cost equal to the sum of the costs of the directed links (u, v) and (v, u) in G. Thus, given a group-shared multicast tree, say T, in G, we can construct the corresponding multicast tree, say T', in G', and vice versa. Now, it is easy to verify that T is a minimum-cost group-shared multicast tree in G if and only if T' is a Steiner tree in G'. Similarly, it can be shown that there is a one-to-one correspondence between a source-specific multicast tree in G which employs bidirectional links and a source-specific multicast tree in G'.

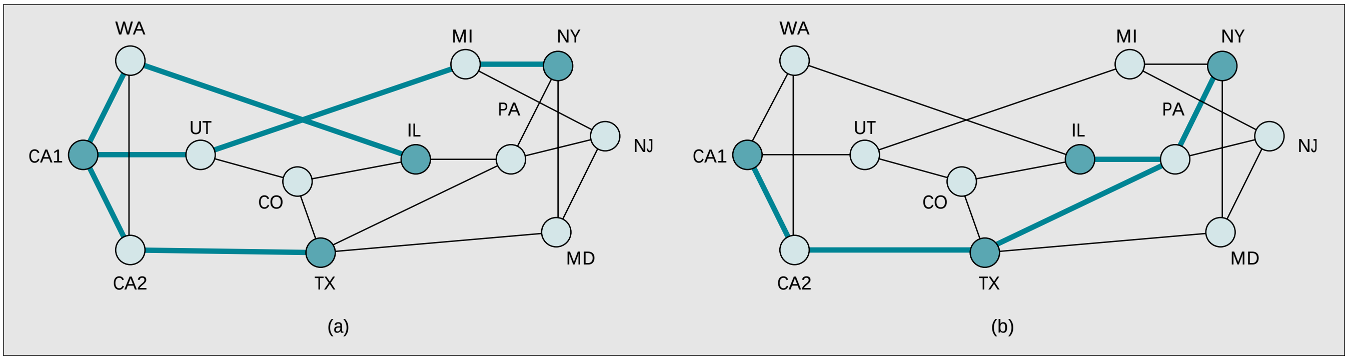
Next, we examine the problem of finding a source-specific multicast tree which employs unidirectional links (note that this problem cannot be reduced to SPN). Such a multicast tree can be modeled using a source-rooted directed Steiner tree (DST), as follows [17].

Let $G = (V, E)$ be a directed graph, C a cost function, M the multicast group, and $s \in M$ the source node. Let *indegree*(v) denote the in-degree of node v, and let *outdegree*(v) denote the out-degree of node v in the directed graph G. Let $T = (V_T, E_T)$ be a DST of G, where $V_T \subseteq V$, $E_T \subseteq E$, and $M \subseteq V_T$. Then a directed path exists in T from s to every node in $M - \{s\}$ such that $\forall v \in V_T - \{s\}$, $indegree(v) = 1$, $indegree(s) = 0$; $\forall v \in V_T - (M - \{s\})$, $outdegree(v) \geq 1$, and the cost of the directed tree $C_T = \Sigma_{(u,v) \in E_T} c_{uv}$ is the minimum of all such directed trees of G. In other words, a DST is a minimum-cost directed tree, rooted at source s, containing the destination nodes $M - \{s\}$ with all links directed away from s.

Recall that in the undirected version of the Steiner tree problem, we were able to develop many simple algorithms which had a constant performance guarantee of 2. For the DST problem, the existence of an approximate algorithm with a constant performance guarantee is as unlikely as $P = NP$ [17]. Thus, the asymmetry in the directed graph prevents us from finding a good approximate solution for the DST problem. We formalize the notion of asymmetry of a graph by defining the maximum link asymmetry as follows [17]:

The *maximum link asymmetry* $\Psi_m(G)$ of a graph $G = (V, E)$ is the maximum ratio of the costs between the two directed links of a link pair. That is,

$$\Psi_m(G) = \max_{(u,v)\in E} \frac{\max(c_{uv}, c_{vu})}{\min(c_{uv}, c_{vu})}. \tag{1}$$



■ Figure 7. *Delay optimization examples: a) a shortest-path tree rooted at the source node CA1; b) an optimum center-based tree rooted at the center node* TX. *The multicast group consists of nodes* (CA1, TX, IL, NY). *The cost of a shortest-path tree is 7, and the average source-specific delay is 2.33. The cost of the center-based tree is 5, and the average group-shared delay is 2.67.*

Now, we describe an approximate algorithm which has a performance guarantee of $2\Psi_m(G)$ [17] and is an adaptation of Prim's spanning tree algorithm [11]. The algorithm grows the DST starting from the source, *s*. At every step, a directed path is extended from the tree constructed so far to one new node $v \in M - \{s\}$ which is not already in the tree. This step is repeated until all of the nodes in *M* are included in the tree. The new node to be added to the DST is selected as follows. First, nodes already in the tree (both multicast and Steiner nodes) are arranged in a priority queue such that *priority*(*source*) > *priority*(*multicast node*) > *priority*(*Steiner node*). No ordering is performed among any two multicast nodes or any two Steiner nodes. Then the first κ nodes in the priority queue are taken and placed in a set *B*. The multicast node *v* that is selected is the one that is closest to *B*. That is, among all directed paths from a node in *B* to a multicast node not in the tree, the path of least cost and the multicast node on which this path terminates are chosen. The parameter κ is a *knob* that can be used to set the algorithm to a desired trade-off between expected tree cost and running time; that is, as we increase κ, the tree-cost decreases but the running time increases.

Delay Optimization

This section studies the problem of optimizing the delay of a multicast tree. Note that the definition of delay for a source-specific multicast tree is different from that for a group-shared multicast tree. If *s* is the source, the average *source-specific delay* (with respect to source *s*), DS_s, of the multicast tree is defined as

$$DS_s = \frac{1}{|M|-1} \sum_{v \in M, v \neq s} P_d(s,v), \tag{2}$$

where $P_d(s, v)$ is the delay of the path from source *s* to multicast node *v* in the multicast tree. Similarly, the average *group-shared delay, DG*, of a multicast tree is defined as

$$DG = \frac{1}{|M|} \sum_{v \in M} DS_v, \tag{3}$$

where DS_v is the average source-specific delay (with respect to source *v*) of the multicast tree.

The problem of finding a source-specific multicast tree which minimizes DS_s has a simple solution, described below. On the other hand, the problem of finding a group-shared multicast tree which minimizes the value of *DG* is NP-complete [18], and will be discussed later in this subsection.

An optimum source-specific delay multicast tree (for both unidirectional and bidirectional link cases) is also called the *shortest-path tree* and is defined as follows. Let s be the source of a source-specific multicast tree and $SP(s, v)$ be the shortest-path from *s* to node $v \in M - \{s\}$. Construct a graph G_{SP} by taking the union of all the shortest paths SP(s, v), where s is the source node and v is a multicast node. Now, the shortest-path multicast tree is obtained by removing all the loops in G_{SP}. In Fig. 7a, the shortest paths from source *CA*1 to destinations *TX*, *IL*, and *NY* are *CA*1-*CA*2-*TX*, *CA*1-*WA*-*IL*, and *CA*1-UT-MI-NY, respectively. Thus, the cost of the shortest-path tree is 7, and it consists of links (*CA*1, *CA*2), (*CA*2, *TX*), (*CA*1, *WA*), (*WA*, *IL*), (*CA*1, *UT*), (*UT*, *MI*), and (*MI*, *NY*), as shown in Fig. 7a.

As mentioned before, finding a multicast tree which optimizes the average group-shared delay (*DG*) is NP-complete, although polynomial-time approximation algorithms exist which have a constant performance guarantee. One such algorithm finds an optimum center-based tree which is defined as follows. Let T_v be the shortest-path multicast tree rooted at node *v* and DG_v^T be the average group-shared delay of T_v. Then an optimum center-based tree is defined as the shortest-path tree with the minimum value of DG_v^T. This tree can easily be found in polynomial time by computing the DG_v^T values for all nodes $v \in G$, and taking the minimum. An optimum center-based tree has a performance guarantee of 2, that is, the average group-shared delay of an optimum center-based tree is guaranteed to be within two times an optimum group-shared delay [19]. Note that the root of the shortest-path tree may not be a multicast member; it may be any node in the graph. If we choose the center only from the multicast members, the performance guarantee of such a center-based tree is 3 [19]. Figure 7b shows an optimum center-based tree. The center is *TX*, and the average group-shared delay is 2.67.

The Cost-Delay Trade-off

In the previous two subsections, we studied algorithms for optimizing the cost and delay of a multicast routing tree. In this subsection we examine the problem of finding a source-specific multicast tree which attempts to optimize *both* cost and delay. In general, a single multicast tree cannot have minimum cost and minimum delay. For example, if *s* is the source of a multicast connection and *T* is the Steiner tree found by the KMB algorithm, the average source-specific delay (with respect to source *s*) of *T* is bounded from above by $(|M| + 1)/2$ times the minimum average source-specific delay [20]. Similarly, the shortest-path tree optimizes the source-specific delay (with respect to source *s*), but it can be $|M|$ times costlier than the Steiner tree, although empirical data suggests that, on an average, the shortest-path tree may only be *slightly* (20 percent) costlier than the Steiner tree found by approximation algorithms such as KMB [20]. On the other hand, the average source-specific delay of an approximation algorithm, such as KMB, is typically larger[7]

[7] *Note that these results assume that every link has the same cost and delay values (i.e., if the cost of a link is* x, *then the delay of the link must also be* x*).*

(50 percent) than the average source-specific delay of the shortest-path multicast tree [21].

Thus, it is natural to ask if an algorithm exists to improve the source-specific delay characteristics of a multicast tree produced by an approximation algorithm such as KMB. One such algorithm, which is described below, is based on the following intuition. Given an optimum-cost (or near-optimum-cost) Steiner tree, if we find the multicast destination for which the delay in the Steiner tree differs the most from the delay of the corresponding shortest path, and connect this destination to the source by the shortest path, we can decrease the average source-specific delay. Thus, if $P_d(s, v)$ is the delay of the path in the optimum-cost (or near-optimum-cost) Steiner tree from source s to multicast destination v, and if $SP(s, v)$ is the shortest-path delay from node s to node v, then the following represents $|M| - 1$ different trade-off algorithms, parameterized by the variable i, where $i = 1, \ldots, |M| - 1$.

- Do i times:
  –Step 1: Find $v \in M - \{s\}$ which maximizes $\mathrm{Pd}(s, v) - \mathrm{SP}(s, v)$.
  –Step 2: Replace the path from s to v in the Steiner tree by the shortest path from s to v.

If we assume that every link has the same cost and delay values, the above method generates Steiner trees within $c\sqrt{|M|}$ of the optimum source-specific delay and cost values for some constant c [20].

Scalability

In order to support multicast applications over large networks, the multicast routing algorithm should be scalable, that is:

- For a network with a large number of nodes, finding a multicast tree should require little time and few resources per node.
- It should be possible for a large number of multicast trees to coexist without requiring an unreasonable amount of routing information at each node.

This subsection examines these two properties of a scalable multicast algorithm.

One method to categorize multicast routing algorithms is as follows:

- Algorithms that require global knowledge of the network topology
- Algorithms that require partial knowledge of the network topology

For example, if we employ the KMB algorithm, each node requires global knowledge of the network topology in order to compute the closure graph. On the other hand, if we employ the center-based-tree algorithm, each node only needs to know the next hop along the shortest path to every destination (which is exactly the information contained in the unicast routing table). In general, algorithms which require global knowledge of the network topology are not as scalable as those which require partial knowledge of the network topology.

The second characteristic of a scalable multicast algorithm is that it should be possible for a large number of multicast trees to coexist without requiring large routing tables at each node. Given that every source and every destination has its own unique address, if the network has a large number of nodes, it is impossible to store routing information for every destination. Hence, to reduce the size of the routing tables, large networks often use a hierarchical addressing scheme. In a hierarchical addressing scheme, routing is performed by inspecting only a portion of the destination address; thus, a single entry in the routing table is sufficient for routing to a large number of destinations. For multicasting, it is difficult (if not impossible) to construct such a hierarchical addressing scheme. Thus, if node A belongs to N multicast trees, the multicast routing table at node A has at least N entries — one entry for each multicast group. On the other hand, a source-specific tree is identified by the tuple (*source_address, multicast_address*). Thus, if node B belongs to M multicast groups, each of which have S sources, its multicast routing table will have at least M x S entries — one entry for each source in each multicast group. Now, since the number of sources in a multicast group may be quite large, a multicast routing protocol based on source-specific trees is not as scalable as a multicast routing protocol based on center-based trees [22, 23].

Dynamic Multicast Groups

Multicast groups are dynamic in nature; that is, new members may join the multicast group and existing members leave at different points in time. Thus, a good multicast routing algorithm should not only allow multicast members to join and leave the multicast tree in a seamless fashion; it should also ensure that a join or leave event does not require widespread changes in the routing tables in the network. Moreover, the quality of a multicast tree (cost, delay, etc.) should not degrade because of a join or leave event.

The following algorithm tries to minimize the cost of a multicast tree for a dynamic multicast group [24]. Let T be a multicast routing tree. First, let us consider a join event. Let u be a node that is to be included in the multicast tree. Let v be a node in the multicast tree, and let d_{uv} be the distance of the shortest path from node u to node v in the network. Let $\omega \in (0, 0.5)$ be a real-valued constant. Also, let each multicast tree contain a special node, say z. Now, connect node u to the node v in the multicast tree which minimizes the value of $(1 - \omega)d_{uv} + \omega d_{vz}$. If $\omega = 0$, then the algorithm is greedy (i.e., it connects node u to the nearest node in the multicast tree), while if $\omega = 0.5$, then the algorithm connects node u to the special node (z) by the shortest path. In case of a leave event, we simply remove the node from the multicast tree if it is a leaf node; otherwise, we remove the node from the multicast group but not from the tree. Empirical results show that a value for ω in the neighborhood of 0.3 yields the best results (i.e., the quality of the multicast tree does not deteriorate even after numerous join and leave events) [24].

Survivability

It is well known that a communication network failure can have an extremely crippling effect on today's society. In the future, as more applications employ multicast routing, a strong need will emerge for algorithms that can be employed by survivable multicast routing protocols.

A survivable routing protocol is designed so that it can survive multiple link (or node) failures; that is, in the event of multiple link (or node) failures, the routing protocol reroutes the connection(s) so as to minimize the networkwide data loss. Unicast routing protocols employ various rerouting algorithms to provide survivability against multiple link (or node) failures. These techniques can be broadly classified into two categories: *protection* and *restoration*. In protection, extra network resources are reserved during the connection setup phase in order to implement survivability. The network resources used for protection may be reserved *separately* for each failure scenario; alternatively, the resources used for protection may be *shared* among different failure scenarios. In restoration, the network resources are dynamically *reassigned* in the event of a failure. Usually packet-switched networks employ restoration to implement survivability. Routing Information Protocol (RIP) and Open Shortest Path First (OSPF) are examples of protocols that implement restoration on IP networks. To the best of our knowledge, we are not aware of any multicast routing protocol that employs an algorithm designed specifically to provide survivability for multicast connections. We believe that this topic needs further research.

Note that multicast routing protocols based on an underlying unicast routing protocol are as survivable as the underlying unicast routing protocol. For example, an implementation of the Distance-Vector Multicast Routing Protocol (DVMRP) may employ unicast routes in the underlying network to perform multicast routing. Thus, in the event of a failure, if the unicast routing tables are updated appropriately, DVMRP will also function correctly. On the other hand, if a multicast routing protocol is independent of the unicast routing protocol, it must implement its own restoration mechanism.

Fairness

Since multiple sources and destinations participate in multicast routing, the issue of fairness arises naturally. In this subsection we consider two fairness issues. The first issue concerns the variation of the delay values from the source to different destinations in a source-specific multicast tree. For example, during a teleconference, it may be important that the speaker be heard by all participants within a bounded time; otherwise, the teleconference may lack the feel of an interactive discussion. Similarly, in a distributed game, the ability to access multicast data before others may result in an unfair competitive advantage. Thus, some applications (e.g., teleconferencing) may impose an upper bound on the delay from the source to a destination in a multicast tree, while other applications (e.g., distributed games) may impose a more stringent condition on the delay: not only should the delay be bounded, but the variation of the delay for different (*source, destination*) pairs should also be bounded. First, we formulate the problem of finding a *delay-bounded Steiner tree* (DBST); then, we formulate the problem of finding a *delay-bounded and delay-variation-bounded multicast tree* (DVBMT).

The problem of finding a DBST can be formulated as follows [25]. Let $G = (V, E)$ be an undirected graph and let $s \in M$ be the source of a source-specific multicast tree. Let the delay of link (u, v) be denoted D_{uv}, a positive integer. Given an integer delay tolerance Δ, a constrained multicast tree T is defined as a tree, rooted at s, that spans the nodes in M such that for each node $t \in M - \{s\}$, the delay on the path from s to t is bounded from above by Δ. Thus, for each $t \in M - \{s\}$, if P_{st} is the path in T from s to t, then $\Sigma_{(u,\, v) \in P_{st}} D_{uv} < \Delta$. Now, the DBST is defined as a constrained multicast tree spanning M such that $\Sigma_{(u,\, v) \in T}\, c_{uv}$ is minimized.

The problem of finding a DVBMT can be formulated as follows [26]. Let $s \in M$ be the source of a source-specific multicast tree, let D_{uv} be the delay of link (u, v), let Δ be the delay tolerance, and let δ be the delay variation tolerance. Now, the DVBMT is a tree T spanning M, such that if $P_d(s, v)$ is the delay of the path from s to v in the multicast tree, then for all $v \in M - \{s\}$, $P_d(s, v) \leq \Delta$, and for all pairs of multicast nodes $u, v \in M - \{s\}$, $|P_d(s, u) - P_d(s, v)| \leq \delta$. In other words, in the multicast tree, the distance from the source to each multicast node should be less than the delay tolerance (Δ) and, for any two multicast nodes (say u and v), the difference between the delay from the source to these two nodes ($|P_d(s, u) - P_d(s, v)|$) should be less than the delay variation tolerance (δ). DVBMT is NP-complete [26]. In [26], a polynomial-time heuristic algorithm is described for DVMBT. Of course, since DVBMT is NP-complete, a polynomial-time algorithm may fail to find a solution even if one exists.

The second fairness issue concerns the data duplication responsibility of a multicast router. Recall that in multicast routing, some routers need to duplicate packets. In a heterogeneous high-speed network, some switches may not have multicast capability. Even if all the switches have multicast capability, by limiting the number of copies made, we may reduce the load at a switch. Limited data duplication at the switches translates to an upper limit on the degree of the nodes in the multicast tree.

The problem of finding an optimum multicast tree such that the degree of every node in the tree is less than a certain value is called the *degree-constrained multicast tree problem* [27], and it can be modeled using the d*egree-constrained Steiner problem in networks* (DCSPN) [28] as follows. Let $\pi(v)$ be the degree constraint at node v, and let deg(v) denote the degree of node v in the degree-constrained multicast tree. Then, for all nodes v in the degree-constrained multicast tree, $\deg(v) \leq \pi(v)$. Thus, given the degree constraint $\pi(v)$ for all nodes in the network, a degree-constrained Steiner tree, T, is a tree which spans all the nodes in M such that $\forall v \in$ T, $\deg(v) \leq \pi(v)$, and the cost of the tree is minimum among all possible multicast trees satisfying the degree constraint.

DCSPN is NP-complete [28]. In fact, finding any multicast tree (not necessarily the minimum-cost multicast tree) which satisfies the degree constraint is an NP-complete problem.

Multicast Routing Protocols

In this section we provide an overview of various multicast protocols employed on the Internet. For a detailed discussion of these and other multicast routing protocols, the reader may refer to some recent books on multicasting [5, 6].

Multicasting over a large portion of the Internet was first demonstrated in March 1992 [29] over the Multicast Backbone (MBone). The Mbone is a virtual network on top of the Internet which provides a multicast facility to the Internet. The MBone can be viewed as a collection of “islands” that support multicasting within their domains. Each island has a host machine which executes the *mrouted* multicast routing daemon. The *mrouted* daemons (in different islands) are connected to one another via point-to-point IP connections (called *tunnels*) over the Internet. In this manner, the mrouted daemons and the tunnels that connect them form a virtual network on top of the Internet.

Multicasting on the Internet is implemented by employing three types of protocols. The first type of protocol is employed by a host to join and leave a multicast group. An example of this type of protocol is the Internet Group Management Protocol (IGMP) [30]. The second type of protocol is called a Multicast Interior Gateway Protocol (MIGP) and is employed by multicast routers to enable multicast communication within an autonomous system (AS).[8] Some examples of MIGPs are DVMRP [31], Multicast Extensions for OSPF (MOSPF) [32, 33], Core-Based Tree (CBT) [22, 23], and Protocol-Independent Multicast (PIM) [34, 35]. The third type of protocol is employed by *border routers*[9] to allow multicast communication across ASes. An example of this type of protocol is the Border Gateway Multicast Protocol (BGMP) [36]. Figure 8 shows how the three types of protocols interoperate in a network (further details on Fig. 8 can be found later).

Of all the multicast routing algorithms discussed in the previous section, only a few are used in practice. DVMRP and MOSPF employ a shortest-path tree,[10] while CBT and BGMP employ a center-based tree to route multicast packets. PIM can employ either a center-based tree or a reverse-shortest-

[8] *An AS is a network of routers under the control of a single administrative domain.*

[9] *Border routers are routers that interconnect two ASes.*

[10] *More precisely, DVMRP employs a reverse shortest-path tree, and MOSPF employs a shortest-path tree.*

path tree (how PIM determines which type of tree to use will be discussed later). We believe that the two main reasons why other, more-sophisticated multicast routing algorithms, such as KMB, are not used are:

- *Ease of implementation*: Recall that the shortest-path tree (or the center-based tree) is composed of two or more shortest paths. Since unicast routing algorithms also compute shortest paths, the multicast routing protocol can be implemented as an add-on to the unicast routing protocol.
- *Efficient computation of the multicast tree*: Note that finding the shortest-path or center-based tree requires much less computation and memory resources than are required by other sophisticated algorithms, such as KMB.

The remainder of this section is organized as follows. We examine IGMP and describe Reverse-Path Multicast (RPM), a multicast routing algorithm employed by DVMRP. We also examine DVMRP, MOSPF, CBT, PIM, and BGMP, respectively. We also briefly examine two new protocols, called EXPRESS [37] and Simple Multicast [38].

Internet Group Management Protocol

IGMP [30] is a protocol that is implemented within the IP module of a host,[11] and it extends the host's IP implementation to support multicasting. IGMP is used between a host and the immediately neighboring multicast router.

Multicast groups are identified by class D IP addresses (i.e., those with 1110 as their high-order four bits). The address 224.0.0.0 is guaranteed *not* to be assigned to any group, and 224.0.0.1 is assigned to the permanent group of all IP hosts (including gateways). This address is used to address all multicast hosts on the directly connected network. There is no multicast address (or any other IP address) for all hosts on the entire Internet.

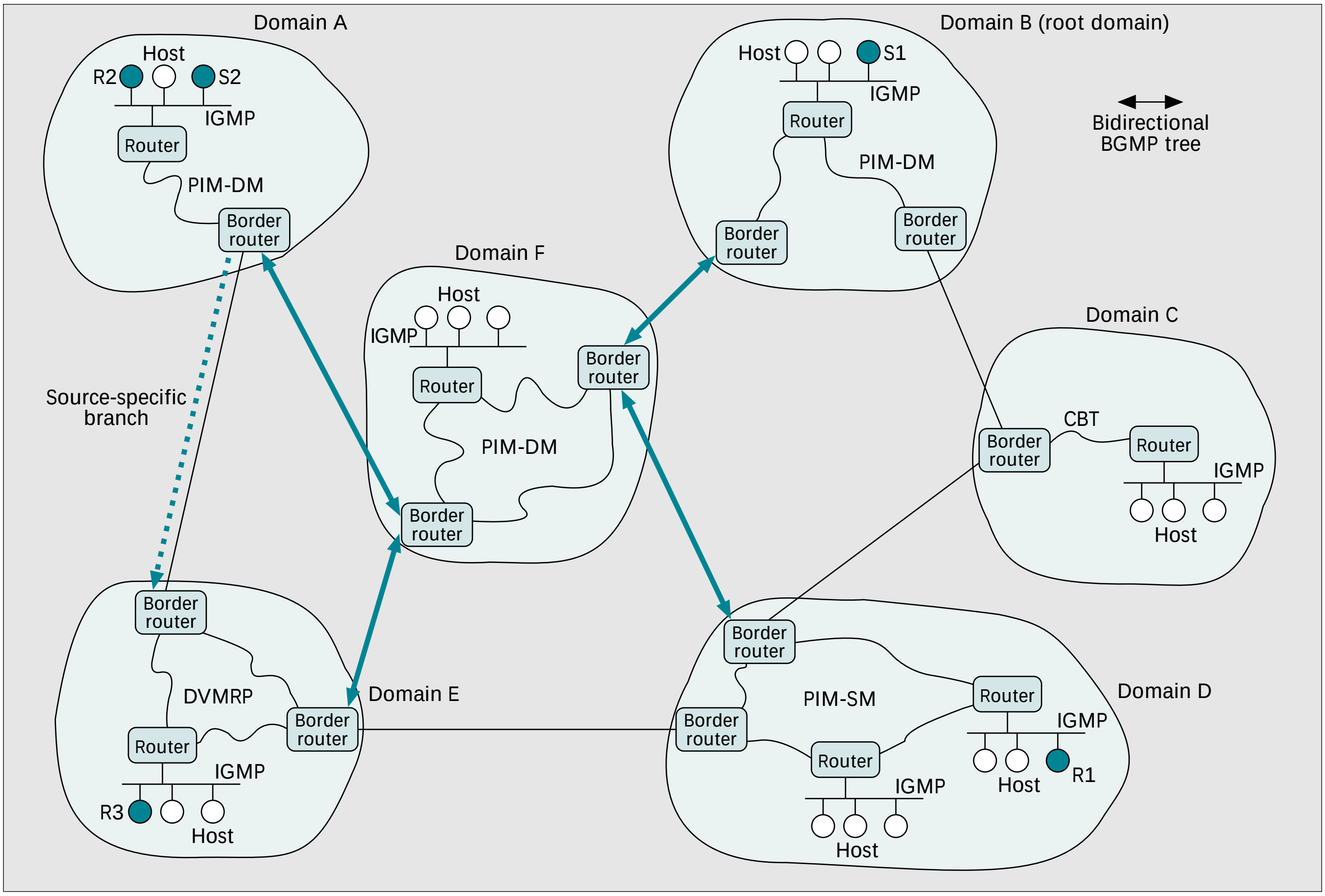
IGMP enables a multicast router to keep track of multicast group membership information by employing two types of IGMP messages: *host membership query* and *host membership report*. Host membership query messages are periodically sent by multicast routers to discover which multicast groups have members on the attached local network. Queries are addressed to the all-hosts group (address 224.0.0.1). Hosts respond to a query by generating host membership reports (hereafter called *join-group reports*), reporting each multicast group to which they belong. When a host joins a new group, it immediately sends a join-group report for that group rather than wait for a query. When a host decides to leave a group, it sends a leave-group report to the multicast router.

Reverse-Path Multicast

RPM [39] enables multicast routing over a network of routers connected to each other via communication links.[12] In order to understand RPM, we must first examine a related protocol called *Reverse-Path Forwarding* (RPF) [40] which broadcasts a packet over a network.

[11] *A host is any Internet host or gateway other than those acting as multicast routers.*

[12] *A communication link may be a regular point-to-point link, a tunnel, or a LAN which is shared by the two routers.*



■ Figure 8. *Various protocols that implement multicasting over the Internet.*

In RPF, a router R forwards a broadcast packet originating at source s if and only if it arrives via the shortest path from the router R back to the source S (i.e., the reverse path). The router forwards the packet on all incident links except the one on which the packet arrived. In this manner, RPF accomplishes a broadcast by flooding packets throughout the network. It should be noted that in RPF, multiple copies of the same packet may be sent over a single link.

TTL threshold	Scope
0	Restricted to the same host
1	Restricted to the same subnetwork
15	Restricted to the same site
63	Restricted to the same region
127	Worldwide
191	Worldwide; limited bandwidth
255	Unrestricted in scope

■ Table 1. *An example of TTL values and their scopes.*

In RPM, three modifications are made to the RPF algorithm:

- Each multicast router R knows its child links, the links with routers whose next-hop router along the shortest path to the source is R. In other words, each link in the network has a unique parent router relative to each possible source S. The set of routers and the corresponding child links form a spanning tree rooted at source S called the *reverse-shortest-path tree* (RSPT). Thus, by employing the RSPT, a source node (S) can broadcast a packet to all the nodes in the network.
- Each multicast router knows whether its subnetwork (i.e., the LAN to which it is connected) has any hosts which are members of a multicast group. Recall that a multicast router collects this information by employing IGMP.
- If a multicast router does not want to receive packets destined to a multicast group, it can send back a prune message to its parent (next-hop router to the source). Note that the prune messages implement *on-demand pruning* of the RSPT.

Thus, RPM consists of two steps. First, a multicast packet is broadcast over the RSPT. When the packet reaches a multicast router for whom none of the child links have members that belong to the multicast address, a prune message for that (*source*, *group*) pair is generated and sent back to the parent multicast router (Fig. 9). When a member of a new group on a particular link appears, a cancellation message to undo the effect of the prune message is sent out by the router.

A prune message includes an *age* field which is initialized by the router that generates the report and increased in value by every router along the reverse shortest path that receives the report. When the age of a prune message reaches a threshold, T_{maxage}, it is discarded. This idea ensures that the prune messages in the network do not contain outdated information.

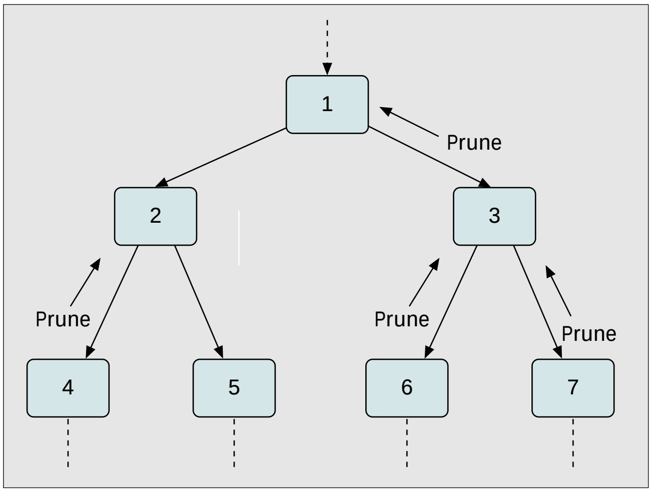
Distance-Vector Multicast Routing Protocol

DVMRP [31] is a multicast routing protocol which employs RPM to send multicast packets over a communication network. DVMRP assigns each communication link a *metric* and a *threshold*. The metric specifies the routing cost of the link and is used for constructing the RSPT. The threshold is the minimum time to live (TTL) a multicast packet needs to be forwarded onto a given link. In this way, the threshold can be used to limit the geographical scope (i.e., region) of a multicast transmission. Table 1 lists some conventional TTL values that are used to restrict the scope of an IP multicast.[13]

In DVMRP, multicast routers periodically exchange routing table update messages with their neighbors. These updates are independent of those generated by any interior gateway protocol, such as RIP, which maintains routing tables for unicasting. Based on the updates from its neighbors, a router builds its multicast routing tables.[14] A sample routing table for a DVMRP router is shown in Table 2. Since a multicast routing table is based on the RSPT, the Source and From gateway columns in Table 2 correspond to the Destination and Gateway columns in a normal unicast routing table.

Multicast Extensions to OSPF

MOSPF [32, 33] multicasts packets over the shortest-path tree within an AS. OSPF [33] is a unicast routing protocol that is employed within an AS. Each OSPF router maintains a database, called the *link state database*, which describes the network topology of the AS. In OSPF, the link state database is constructed using five different types of link-state advertisements (LSAs). An LSA is a "unit of data describing the local state of a router or network. For a router, this includes the state of the router's interfaces and adjacencies. Each link state advertisement is flooded throughout the routing domain" [33]. MOSPF extends OSPF by adding a new type of LSA, called the *group membership LSA*. In MOSPF, a router uses IGMP to keep track of group membership information on its attached network, and distributes this information by flooding the group membership LSA throughout the AS. Thus, by employing the link state database, a router can compute a shortest-path tree for any node in the AS. When a router receives a multicast packet, it computes a shortest-path tree rooted at the source of the packet and forwards the packet accordingly. In order to conserve CPU and memory resources at the router, the shortest-path tree is computed on demand (i.e., at the arrival of the first multicast packet).



■ Figure 9. *An example of how a prune message is generated and sent to the parent router. Nodes 6 and 7 send a prune message to their parent node (3), which in turn generates a prune message and sends it to its parent node (1). Node 2 does not generate a prune message because it received a prune message from only one of its two child nodes.*

[13] *Since TTL-based scoping is coarse-grained, it may not be appropriate for certain applications. In such cases, we may employ Administratively Scoped IP Multicast [41, 42] which is a more fine-grained scoping method than the TTL-based method.*

[14] *The algorithm employed in DVMRP to build routing tables is very similar to the one employed by RIP. For further details, please refer to [31, 43].*

Source	Subnet mask	From gateway	Metric	Status	TTL
128.1.0.0	255.255.0.0	128.7.5.2	3	Up	200
128.2.0.0	255.255.0.0	128.7.5.2	5	Up	150
128.3.0.0	255.255.0.0	128.6.3.1	4	Up	200

Table 2. *An example of a DVMRP routing table.*

Core-Based Tree (CBT)

In a very large network with many simultaneously active multicast groups, DVMRP can become very costly for two reasons. First, the broadcast of the initial packet in RPM can be costly if the network consists of tens of thousands of nodes, of which only a few are a part of the multicast group. Second, each multicast router has to keep track of every (*source*, *group*) pair, which may become unwieldy as the number of multicast groups and sources increase. The CBT architecture [22, 23] is an attempt to overcome the shortcomings of DVMRP. In CBT, branches emanate from a single node known as the core of the tree. These branches are made up of other routers, so-called *on-tree* routers, which form a shortest path between a host's directly attached router and the core. The CBT architecture significantly decreases the size of multicast routing tables at the routers, because it requires the routers to store routing information for every active group (i.e., per tree) as opposed to storing information for every active (*source*, *group*) pair. Once the core router is chosen, routers that are not on the CBT can send a JOIN\REQUEST message to the core router which sets up the routing tables at every hop. In this manner, CBT creates a bidirectional shared center-based tree.

Protocol-Independent Multicast

In attempting to remove the shortcomings of DVMRP, CBT inadvertently introduces some new problems. In [34] CBT's shortcomings were analyzed, and a new protocol, called PIM [34, 44, 45], was presented, which addresses these shortcomings. To understand the motivation behind PIM, we must first understand the limitations of CBT.

As seen in the previous section, CBT uses a single delivery tree for each group, routed at a core router and shared by all nodes which send packets to the multicast destination set. As desired for sparse groups, CBT does not exhibit the occasional broadcasting behavior of RPM. However, CBT does so at the expense of imposing a single shared tree for each multicast group. This can result in concentration of all the sources' traffic on a single link. In [34], this phenomenon is referred to as *traffic concentration*. This is one of the limitations of CBT, or any protocol that imposes a single shared tree per group for distribution of all data packets.

It is evident, though, that both types of trees (RSPTs and CBTs) have their advantages. For example, shared trees may perform very well for a large number of low-data-rate sources which are spread over a large geographical area (e.g., resource discovery applications), while RSPTs may be better suited for high-data-rate sources (e.g., real-time videoconferencing). An analysis of these trade-offs can be found in [21]. It would be ideal to flexibly support both types of trees within one multicast architecture, so the selection of tree types becomes a configuration decision within a multicast protocol.

PIM is designed to address the two issues stated above: to avoid the overhead of broadcasting packets when group members sparsely populate the Internet, and to do so in a way that supports good-quality distribution trees for heterogeneous applications. Thus, PIM has two modes of operation: PIM Dense Mode (PIM-DM), which employs an RSPT (similar to DVMRP), and PIM Sparse Mode (PIM-SM), which employs unidirectional shared trees. PIM-DM multicast tree construction is data-driven, that is, dense mode will be used only if data rates exceed a certain value.

Some of the important facts about PIM-SM are as follows:

- PIM-SM employs per-group *rendezvous points* (RPs) for receivers to meet new sources. RPs are used by senders to announce their existence, and by receivers to learn about new senders of a group.
- Routers with local (or downstream) members join a PIM-SM tree using *explicit* join messages. (In contrast, DVMRP generates the multicast tree by pruning an RSPT.)

Now we explain how a host can join a group and receive multicast packets using PIM-SM. We assume that routers listen to a well-known multicast group to obtain the group-address-to-RP bindings. Thus, every router knows the designated RPs for a given multicast group. When a host signals that it wants to join a PIM-SM multicast group (i.e., by sending an IGMP message), its first-hop router sends a PIM-join message toward the RP advertised for the group. Processing of this message by intermediate routers sets up the multicast tree branch from the RP to the host. When a source starts to send data packets to a multicast group, it sends a PIM register message, piggybacked on the data packet, to the RPs for that group. The RP responds by sending a join toward the source. Processing of these messages by intermediate routers sets up a packet delivery path from the source to the RP.

If source-specific (i.e., RSPT) distribution trees are desired, a router sends a PIM-join message toward the source. Figure 10 shows the steps involved in joining a multicast group and setting up a source-specific distribution tree. A router can send a PIM prune message to tear down a connection. A router may want to tear down a connection because it is no longer a part of a multicast group, or it has a RSPT connection to the source and does not need the connection to the RP anymore. Note that when a router joins the source through an RSPT, it effectively changes the path for all its downstream routers and hosts.

Border Gateway Multicast Protocol

Border routers employ BGMP [36] to facilitate multicast communication across different ASes. BGMP consists of two components, namely, the *MIGP component* and the *BGMP component*. The border router employs the MIGP component to participate in the MIGP protocol within the AS, and the BGMP component to construct a bidirectional center-based tree with other border routers. In BGMP, the root of the center-based tree is an entire AS rather than a single router. The root AS of a multicast address is the AS which has claimed the multicast address by employing a global multicast address allocation protocol such as the Multicast Address Set Claim (MASC) [46] protocol.

BGMP uses TCP as its transport protocol. Border routers set up a TCP connection between themselves, and exchange BGMP messages. When group memberships change, border routers send incremental join/prune updates to one another. Since the shortest path from a multicast source to a destination can be different than the path imposed by the shared tree, BGMP also allows a border router to attach a source-specific branch to the center-based tree.

Figure 8 demonstrates how BGMP enables multicasting across ASes. Shown in the figure are six ASes (or domains), each of which employs a different MIGP. Consider a multicast group consisting of sources S1 and S2 in domains B and A, respectively, and three multicast receivers R1, R2, and R3 in domains D, A, and E, respectively. We assume that domain B

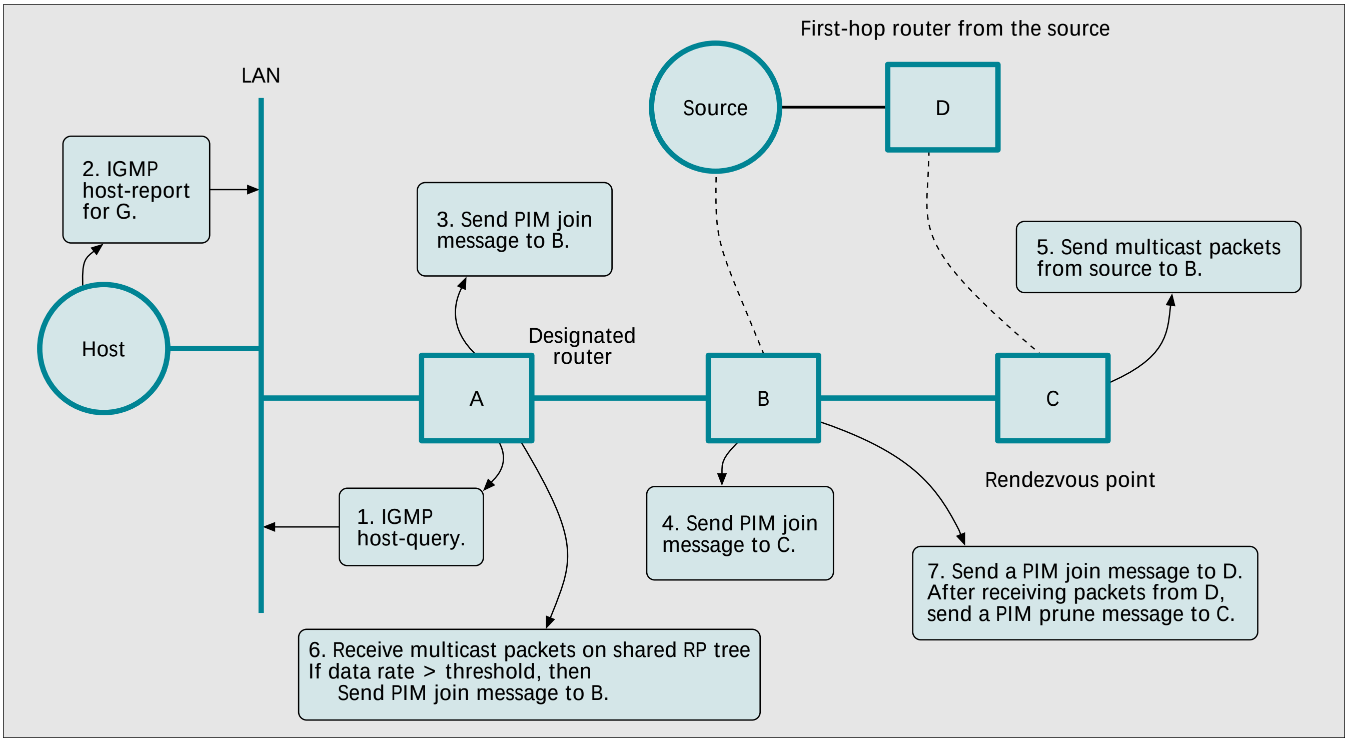

First-hop router from the source
LAN
Source
D
2. IGMP host-report for G.
3. Send PIM join message to B.
5. Send multicast packets from source to B.
Host
Designated router
A
B
C
Rendezvous point
1. IGMP host-query.
4. Send PIM join message to C.
7. Send a PIM join message to D. After receiving packets from D, send a PIM prune message to C.
6. Receive multicast packets on shared RP tree If data rate > threshold, then Send PIM join message to B.

Figure 10. *An xample of how a receiver joins, sets up a shared tree, and switches to a source-specific tree. Actions are numbered in the order in which they occur.*

is the root domain of the multicast group. Thus, the center-based tree created by BGMP for the multicast group is rooted at domain B and is shown by thick bidirectional lines in the figure. Since domain C does not have any nodes belonging to the multicast group, it is not included in the bidirectional multicast tree. Note that multicast packets originating from source B traverse domain F in order to reach domains A and E. Packets originating from source S2 in domain A reach receiver R3 in domain E via the border router in domain F. Since border routers in domains A and E are directly connected, receiver R2 can set up a source-specific branch via the shortest path from source S2 to receiver R2 as shown by the dashed line. Now, receiver R2 will receive packets from source S2 via the source-specific branch, and from other sources via the center-based tree that was set up by BGMP.

Recent Trends

In this section we briefly discuss two recently proposed protocols which extend the IP multicast model. In the current IP multicast model, a multicast address (class D address) refers to a group of hosts. Some of the problems with this model are:

- There is no mechanism to estimate the multicast group size.
- There is no mechanism to restrict unauthorized senders (receivers) from sending (receiving) traffic to (from) the multicast group.
- The model requires a protocol such as MASC to allocate globally unique multicast addresses.

The two protocols discussed in this section attempt to address the above problems by extending the IP multicast model.

Explicitly Requested Single Source (EXPRESS) [37] extends IP multicast to support the *channel* model. A channel consists of one explicitly designated source and zero or more *subscribers*. EXPRESS builds source-specific trees for each channel which are addressed by the tuple (*C*, *M*), where *C* is the source's IP address and *M* is a multicast address. Note that EXPRESS does not require a protocol to allocate globally unique multicast addresses, because a channel is identified not only by the multicast address, but also by the IP address of the source node. Unauthorized hosts can be restricted by associating a key with a channel which is initialized by the source. EXPRESS also provides a mechanism for counting the number of receivers in a multicast group.

Simple Multicast [38] addresses the problem of allocating a globally unique multicast address to each group. It proposes that each multicast group be referred to by the tuple (C, M), where C is the address of the core router of the multicast tree and M is a multicast address. Simple Multicast builds bidirectional shared trees rooted at the core node. Global address management is not an issue because a multicast group is identified not only by the multicast address (class D address), but also by the IP address of the core node.

Conclusion

Today, many multicast applications exist, but the implementation of these applications is not necessarily efficient because today's WANs were designed to mainly support point-to-point (unicast) communication. In the future, as multicast applications become more popular and bandwidth-intensive, there will emerge a pressing need to provide efficient multicast support on WANs. In this work we present a tutorial-cum-survey of some of the important topics in multicasting. First, we study the problem of multicast routing algorithms which are fundamental to all of the research in multicasting, such as *what is a multicast tree, and how does one construct it?* We enumerate the properties of a good multicast tree, and note that finding such a multicast tree can be very difficult. Since most algorithms that have been proposed in the literature mainly focus on optimizing one of the properties of a good multicast tree, we categorize the algorithms based on the property they attempt to optimize and separately examine each category, *viz.* low cost, low delay, scalability, support for dynamic multicast groups, survivability, and fairness.

Next, we examine various protocols that are employed on the Internet, namely, Internet Group Management Protocol (IGMP) [30], Distance-Vector Multicast Routing Protocol

(DVMRP) [31], Multicast Extensions for OSPF (MOSPF) [32, 33], Core-Based Tree (CBT) [22, 23], Protocol-Independent Multicast (PIM) [34, 35], and Border Multicast Gateway Protocol (BGMP) [36]. We also briefly examine recently proposed protocols, namely, Explicitly Requested Single Source (EXPRESS) [37] and Simple Multicast [38].

Research in multicasting covers a very wide range of topics. In this tutorial we cover the topics we believe are most relevant to a general networking audience; thus, additional topics such as reliable multicast, multicast support for mobile computing, layered encoding techniques for multicast audio and video applications, multicast in optical networks, and multicast address management were not covered. Future research topics, solutions to which will be very desirable, include secure group communication, survivability in multicast routing, and congestion control in reliable multicast protocols.

References

[1] P. Winter, "Steiner Problem in Networks: A Survey," *Networks*, vol. 17, Summer 1987, pp. 129–67.

[2] C. Diot, W. Dabbous, and J. Crowcroft, "Multipoint Communication: A Survey of Protocols, Functions, and Mechanisms," *IEEE JSAC*, vol. 15, Apr. 1997, pp. 277–90.

[3] G. Carle and E. W. Biersack, "Survey of Error Recovery Techniques for IP-Based Audio-Visual Multicast Applications," *IEEE Network*, vol. 11, Nov./Dec. 1997, pp. 24–36.

[4] K. Obraczka, "Multicast Transport Protocols: A Survey and Taxonomy," *IEEE Commun. Mag.*, vol. 36, Jan. 1998, pp. 94–102.

[5] S. Paul, *Multicasting on the Internet and Its Applications*, Boston, MA: Kluwer, 1998.

[6] C. K. Miller, *Multicast Networking and Applications*, Reading, MA: Addison-Wesley, 1999.

[7] R. M. Karp, *Complexity of Computer Computations*, New York: Plenum, 1972, pp. 85–103.

[8] R. E. Bellman, "On a Routing Problem," *Qtrly. Applied Math.*, vol. 16, 1958, pp. 87–90.

[9] E. W. Dijkstra, "A Note on Two Problems in Connection with Graphs," *Numer. Math.*, vol. 1, 1959, pp. 269–71.

[10] J. B. Kruksal, "On the Shortest Spanning Subtree of A Graph and the Traveling Salesman Problem," *Proc. Amer. Math. Soc.*, vol. 7, 1956, pp. 48–50.

[11] R. C. Prim, "Shortest Connection Networks and Some Generalizations," *Bell Sys. Tech. J.*, vol. 36, 1957, pp. 1389–1401.

[12] A. Z. Zelikovsky, "An 11/6-Approximation Algorithm for the Network Steiner Problem," *Algorithmica*, vol. 9, May 1993, pp. 463–70.

[13] L. Kou, G. Markowsky, and L. Berman, "A Fast Algorithm for Steiner Trees," *Acta Informatica*, vol. 15, no. 2, 1981, pp. 141–45.

[14] J. Plesnik, "A Bound for the Steiner Tree Problem in Graphs," *Mathematica Slovaca*, vol. 31, no. 2, 1981, pp. 155–63.

[15] V. J. Rayward-Smith and A. Clare, "On Finding Steiner Vertices," *Networks*, vol. 16, Fall 1986, pp. 283–94.

[16] H. Takahashi and A. Matsuyama, "An Approximate Solution for the Steiner Problem in Graphs," *Mathematica Japonica*, vol. 24, no. 6, 1980, pp. 573–77.

[17] S. Ramanathan, "Multicast Tree Generation in Networks with Asymmetric Links," *IEEE/ACM Trans. Net.*, vol. 4, Aug. 1996, pp. 558–68.

[18] D. S. Johnson, J. K. Lenstra, and A. H. G. R. Kan, "The Complexity of the Network Design Problem," *Networks*, vol. 8, Winter 1978, pp. 279–85.

[19] D. W. Wall, "Mechanisms for Broadcast and Selective Broadcast," Ph.D. thesis, Stanford Univ., June 1980.

[20] K. Bharath-Kumar and J. M. Jaffe, "Routing to Multiple Destinations in Computer Networks," *IEEE Trans. Commun.*, vol. COM-31, Mar. 1983, pp. 343–51.

[21] L. Wei and D. Estrin, "The Trade-Offs of Multicast Trees and Algorithms," *Proc. Int'l. Conf. Comp. Commun. Networks*, Sept. 1994.

[22] T. Ballardie, P. Francis, and J. Crowcroft, "Core Based Trees (CBT): An Architecture for Scalable Inter-Domain Multicast Routing," *Comp. Commun. Rev.*, vol. 23, Oct. 1993, pp. 85–95.

[23] T. Ballardie, "Core Based Trees (CBT version 2) Multicast Routing," RFC 2189, Sept. 1997.

[24] B. M. Waxman, "Routing of Multipoint Connections," *IEEE JSAC*, vol. 6, Dec. 1988, pp. 1617–22.

[25] V. P. Kompella, J. C. Pasquale, and G. C. Polyzos, "Multicast Routing for Multimedia Communication," *IEEE/ACM Trans. Net.*, vol. 1, June 1993, pp. 286–92.

[26] G. N. Rouskas and I. Baldine, "Multicast Routing with End-To-End Delay and Delay Variation Constraints," *IEEE JSAC*, vol. 15, Apr. 1997, pp. 346–56.

[27] F. Bauer and A. Varma, "Degree-Constrained Multicasting in Point-To-Point Networks," *Proc. IEEE INFOCOM '95*, vol. 4, Apr. 1995, pp. 369–76.

[28] S. Voss, "Problems with Generalized Steiner Problems," *Algorithmica,* vol. 7, no. 2, 1992–1993, pp. 333–35.

[29] H. Eriksson, "MBone: the Multicast Backbone," *Commun. ACM*, vol. 37, Aug. 1994, pp. 54–60.

[30] S. E. Deering, "Host extensions for IP multicasting," RFC 1112, Aug. 1989.

[31] D. Waitzman and C. Partridge, "Distance Vector Multicast Routing Protocol," RFC 1075, Nov. 1988.

[32] J. Moy, "Multicast Routing Extensions for OSPF," *Commun. ACM*, vol. 37, Aug. 1994, pp. 61–66.

[33] J. Moy, "OSPF Version 2," RFC 2328, Apr. 1998.

[34] S. Deering *et al.*, "The PIM Architecture for Wide-Area Multicast Routing," *IEEE/ACM Trans. Net.*, vol. 4, Apr. 1996, pp. 153–62.

[35] D. Estrin *et al.*, "Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol specification," RFC 2362, June 1998.

[36] D. Thaler, D. Estrin, and D. Meyer, "Border Gateway Multicast Protocol (BGMP): Protocol Specification," Internet draft, Aug. 1998.

[37] H. Holbrook and D. Cheriton, "IP Multicast Channels: EXPRESS Support for Large-Scale Single-Source Applications," *Proc., ACM SIGCOMM '99*, Sept. 1999.

[38] R. Perlman *et al.*, "Simple Multicast: A Design for Simple, Low-Overhead Multicast," Internet draft, Feb. 1999.

[39] S. E. Deering and D. R. Cheriton, "Multicast Routing in Datagram Internetworks and Extended LANs," *ACM Trans. Comp. Sys.*, vol. 8, May 1983, pp. 85–110.

[40] Y. K. Dalal and R. M. Metcalfe, "Reverse Path Forwarding of Broadcast Packets," *Commun. ACM*, vol. 21, pp. 1040–48, Dec. 1978.

[41] D. Meyer, "Administratively Scoped IP Multicast," RFC 2365, July 1998.

[42] D. Thaler, M. Handley, and D. Estrin, "The Internet Multicast Address Allocation Architecture," Internet draft, Oct. 1999.

[43] C. Hedrick, "Routing Information Protocol," RFC 1058, June 1988.

[44] S. Deering *et al.*, "Protocol Independent Multicast-Dense Mode (PIM-DM): Protocol Specification," Internet draft, June 1999.

[45] D. Estrin *et al.*, "Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification," Internet draft, Oct. 1999.

[46] D. Estrin *et al.*, "The Multicast Address-Set Claim (MASC) Protocol," Internet draft, Aug. 1999.

Biographies

LAXMAN SAHASRABUDDHE (sahasrab@cs.ucdavis.edu) received a B.Tech. degree from the Indian Institute of Technology, Kanpur, in 1992, and an M.Tech. degree from the Indian Institute of Technology, Madras, in 1994. He is currently a research assistant with the Networks Research Laboratory at the University of California, Davis, where he is working toward a Ph.D. degree. His research interests include architectures and protocols for WDM local-area and wide-area optical networks.

BISWANATH MUKHERJEE [M] (mukherjee@cs.ucdavis.edu) received a B.Tech. (Hons) degree from the Indian Institute of Technology, Kharagpur, in 1980 and a Ph.D. degree from the University of Washington, Seattle, in June 1987. At Washington he held a GTE Teaching Fellowship and a General Electric Foundation Fellowship. In July 1987 he joined the University of California, Davis, where he has been a professor of computer science since July 1995, and chair of the Computer Science Department since September 1997. He is co-winner of paper awards presented at the 1991 and 1994 National Computer Security Conferences. He serves on the editorial boards of *IEEE/ACM Transactions on Networking*, *IEEE Network*, *ACM/Baltzer Wireless Information Networks* (WINET), *Journal of High-Speed Networks*, and *Photonic Network Communications*. He served as Technical Program Chair of IEEE INFOCOM '96. He is author of the textbook *Optical Communication Networks* (McGraw-Hill, 1997). His research interests include lightwave networks, network security, and wireless networks.