REVIEW ARTICLE

Software Multicast Techniques: A Survey

B.D. Birchler[1], A. Esfahanian[1] and E. Torng[1]

A common communication pattern that arises in many applications in parallel systems is multicast, or one-to-many communication. In multicast, a single node in the system sends a message to multiple destinations in the system. Due to the complexity of implementing multicast in hardware, most machines offer only unicast, or one-to-one communication, in hardware. As a result, multicast must be implemented in software. This paper provides a survey of software multicast techniques for parallel systems that use a direct network interconnection structure. Various routing schemes and switching techniques and their impact on the design of multicast algorithms are discussed. In particular, three communication models have been defined for direct network systems: neighbor-switching, line-switching and node-switching. The existing multicast implementations under each model are described and open problems are discussed.

INTRODUCTION

The grand challenges in scientific computing, e.g., pharmaceutical design, ocean modeling and viscous fluid dynamics, can only be solved by computers that can provide Teraflops of computing power. Parallel computers are considered the most promising technology to solve such problems [1]. Because many (hundreds to thousands) processors can be used to solve problems, efficient communication among the nodes of a multicomputer is a vital component of system performance. One communication pattern that arises in parallel systems is multicast, or one-to-many communication. In multicast, one processor, called the source, sends a single message to multiple destinations in the system. Unicast and broadcast are special cases of multicast. Unicast, also called one-to-one communication, involves a single source and a single destination. In broadcast, or one-to-all communication, a message is sent to all of the nodes in the network. Multicast communication is found in various applications such as parallel search algorithms [2], graph algorithms [3] and barrier synchronization [4]. Multicast is also used for cache-coherency protocols in distributed shared memory systems [5] and it has been defined in the MPI (Message Passing Interface) standard [6] as an essential operation for implementing other collective communication operations such as all-to-all gather and global reduction in distributed memory machines. Because multicast communication is a component of many applications in parallel systems, efficient multicast implementation has been the subject of much study.

This paper addresses multicast communication paradigms for an important class of parallel systems, namely those that use a direct network interconnection structure [7]. Due to the complexity of hardware implementations

1. Department of Computer Science, Michigan State University, East Lansing, MI 48824-1027, USA.

Scientia Iranica, Vol. 3, No. 4, © Sharif University of Technology, January 1997.

of one-to-many communication, most existing direct network systems only support unicast communication in hardware. Thus, multicast must be provided in software. A common software technique is Unicast-Based Multicast (UBM), in which multicast is accomplished by issuing multiple hardware unicast messages. The multicast implementation consists of a number of time steps in which nodes that have previously received the multicast message send the message to uninformed nodes. Ideally, the multicast implementation should maximize the amount of parallel communication; that is, all the informed nodes should send messages during a single time step in order to reduce the total amount of time needed to inform all destinations. In this paper, a survey of software multicast techniques for various systems based on direct networks is presented and a more detailed description of the typical direct network is given. Then, the theoretic framework in which multicast has been studied is discussed. In particular, a graph theoretic model of direct network architectures and various system constraints that influence the multicast implementation are discussed. Moreover, the multicast problems and solutions for the various system models that have been considered are addressed. Finally, the paper is concluded by a list of open problems.

DIRECT NETWORK MODEL

Architecture

Multicast techniques have been widely studied for parallel systems based on a direct network interconnection structure. Many direct network systems (for example, the Intel Touchstone DELTA and the Intel Paragon [1]) use the generic node architecture shown in Figure 1 [8]. Each node consists of a local processor and a separate router to handle communication-related tasks. This allows simultaneous computation and communication within each node [7]. Each router has several input and output channels. Consequently, the router can transmit several distinct messages simultaneously. Normally, each input channel is paired with

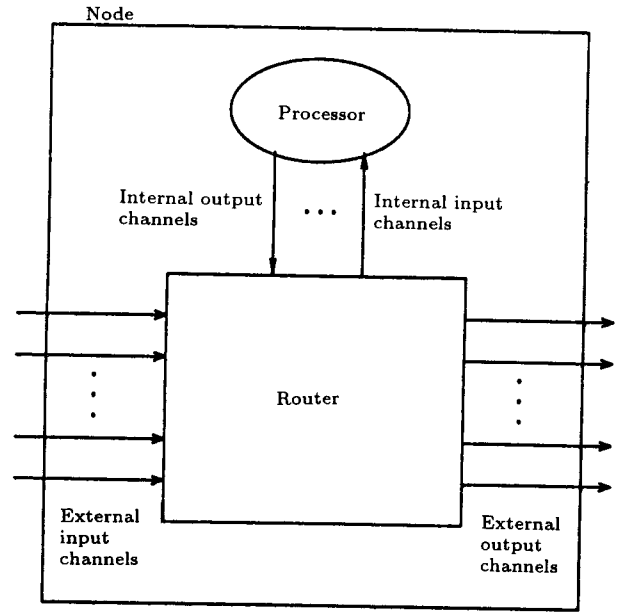

Figure 1. Generic node architecture.

a corresponding output channel. The processor communicates with its router via internal channels. If there is only one pair of internal channels, the system is called a one-port architecture; this survey will focus on multicast algorithms developed for one-port architectures in which a node can send only one message at a time and receive only one message at a time as well. External channels are used for communication between nodes. In the direct network organization, communication is either direct or indirect. Two nodes can communicate directly if an external output channel of one node is connected to an external input channel of the other node, otherwise communication must be done indirectly. For example, in Figure 2 node 1 can communicate directly with node 2. In contrast, sending a message from node 1 to node 3 requires indirect communication.

Popular direct network topologies such as meshes, hypercubes and tori have the property that if node x has an external output channel connected to an external input channel of node y, then y also has an external output channel connected to an external input channel of node x. This allows symmetric (or "two-way") communication between nodes x and y. In an arbitrary topology that uses the generic node architecture, such as the network

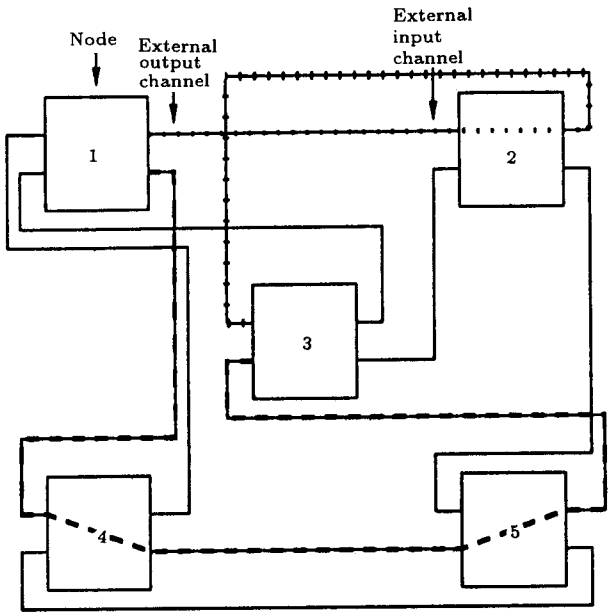

Figure 2. A direct network.

shown in Figure 2, this may not be the case. A multicast algorithm that does not rely on any specific properties or topologies is more powerful; therefore, an arbitrary topology will be assumed unless otherwise stated.

Routing Schemes

In the network shown in Figure 2, node 1 can send a message to node 3 using node 2 as an intermediate node, or it could also use both 4 and 5 as intermediate nodes to deliver a message to node 3. The path along which the message is delivered depends on the routing scheme used. A routing scheme R of a direct network is a collection of all permissible paths along which a message can be delivered for each pair of nodes in the network.

If a routing scheme R includes every (i,j)-path for each v_i and v_j, the system is said to use free routing. If, on the other hand, R includes exactly one (i,j)-path for every pair of nodes v_i and v_j, the system is said to use oblivious routing [9] or restricted routing [8]. Here, the more established term, oblivious routing, will be used. In practice, parallel systems use oblivious routing rather than free routing. For example, current systems based on mesh and hypercube topologies use oblivious routing called dimension-ordered routing. In xy-routing, the

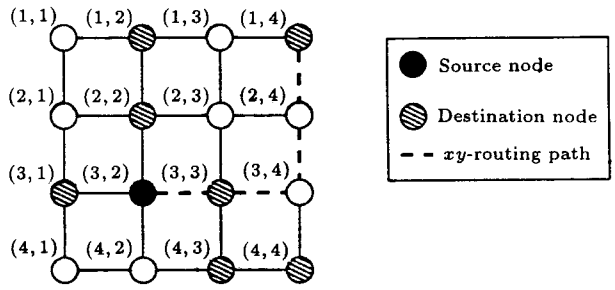

Figure 3. Routing in a 4×4 mesh.

dimension-ordered routing typically used in a two-dimensional mesh, a message moves first in the x direction until it arrives at the correct x coordinate and then in the y direction until it reaches its destination. For example, to deliver a message from node $(3,2)$ to node $(1,4)$ in Figure 3 the path shown by dashed lines is used. The xy-routing scheme is oblivious because the specified path is unique and is used regardless of network conditions. The Intel Paragon [10], a multiprocessor computer with a two dimensional mesh network, utilizes xy-routing. Hypercube topologies typically use e-cube routing, in which the message also travels through the cube dimensions one at a time. The NCUBE-2 [11], a multiprocessor computer with a hypercube network, utilizes e-cube routing.

Switching Techniques

Once a route has been chosen, the time to deliver a message along the routing path depends on the underlying switching mechanism used by the network. One factor that contributes to the message transmission time is the network latency, or elapsed time between when the head of the message enters the network and the time when the tail of the message is received. Network latency is highly dependent on the switching mechanism used. Essentially two types of switching have been used: store-and-forward switching and cut-through switching.

Store-and-Forward Switching

Early direct networks used store-and-forward switching techniques. In a system that uses store-and-forward switching, an entire message is transferred from a node v to one of its

neighbors w. If the message is not destined for w, i.e., w is an intermediate node, the message is forwarded to one of w's neighbors when a buffer becomes available. The network latency for the store-and-forward switching technique is $(\frac{L}{b})d$, where L is the message length, b is the channel bandwidth and d is the distance (number of links) the message must traverse. Because each message must be entirely received at a node before it can be forwarded, the amount of time needed to deliver the message is proportional to the path length d. For a more detailed discussion of routing algorithms based on store-and-forward switching see [12].

Cut-Through Switching

Cut-through switching was developed to reduce message transmission time. A cut-through scheme supports non-local communication primitives; that is, a node v can send a message to a non-neighboring node w in "unit" time. This is possible because messages are not stored at intermediate nodes unless the next required channel is unavailable. The message is sent along the path in a pipelined fashion. The network latency for cut-through switching is $(\frac{L_h}{b})d + \frac{L}{b}$, where L_h is the length of the header, b is the channel bandwidth, d is the length of the path traversed and L is the total length of the message. If L_h is significantly smaller than L, then the dominant term is $\frac{L}{b}$. That is, network latency is relatively unaffected by d. In a multicast implementation, each node is sending essentially the same message. (The message headers may be slightly different, but this will not significantly affect the message length.) Thus, it can be assumed that any (reasonably sized) multicast message can be delivered in unit time, where the "unit" is $\frac{L}{b}$, regardless of the length of the path traversed by the message [7].

Two common routing schemes that use the cut-through approach are circuit routing and wormhole routing. In circuit routing, before a node v can send a message to a node w, v establishes a circuit, or a path, from v to w. The message is then sent along this path in a pipelined fashion. Because the channels on the circuit are reserved, no buffers are needed at intermediate nodes.

In wormhole routing, a packet is divided into a number of flits (flow control digits). The header flit advances along the delivery path and the other flits follow in a pipelined fashion. If the header is blocked, then all trailing flits are also blocked. Consequently, small flit buffers are needed at intermediate nodes. Current commercial multicomputers such as the Cray T3D, NCUBE-2 [11], Intel Paragon [10] and IBM SP-2 [13] use wormhole routing. Ni and McKinley provide a thorough survey of wormhole routing techniques [7].

THEORETIC FRAMEWORK FOR MULTICAST

Typically, the multicast problem has been formulated in a graph theoretic framework. Researchers use a graph to model the parallel system and to develop efficient multicast algorithms. However, before multicast algorithms can be developed for a multiprocessor system, the constraints of the system must be clearly defined and then incorporated into the model.

Graph Theoretic Formulation

Direct networks have been modeled using a graph or a directed graph (digraph for short). In the graph model, a direct network is represented by a graph $G = (V, E)$, where V is the set of nodes in the network and E is the set of the network's communication channels. Each edge $e = v_i v_j \in E(G)$ represents a physical communication link between node v_i and node v_j. In particular, v_i and v_j can send messages to each other along the edge e. The graph G is assumed to be connected and simple, i.e., there are no loops or parallel edges.

In the direct network described in the last section, each of the physical channels represents a one-way communication link. Consequently, communication is not necessarily symmetric, i.e., node i can communicate directly to node j, but node j cannot communicate directly to node i. Networks that are not symmetric are best modeled by digraphs. For example,

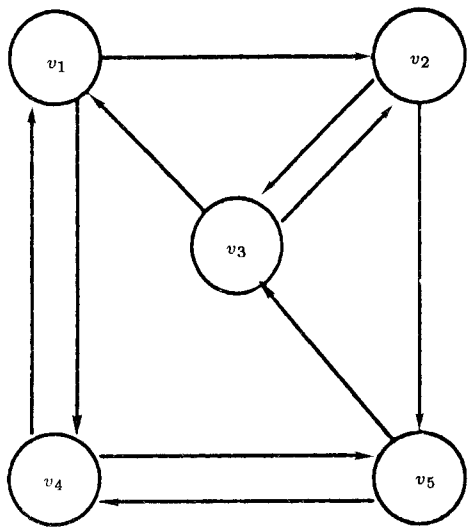

Figure 4. Digraph model of a non-symmetric direct network.

the digraph in Figure 4 models the network pictured in Figure 2. Most popular direct network topologies (e.g., meshes, hypercubes and tori) are symmetric and can be modeled by undirected graphs. A general graph model is used to introduce the main concepts and directed graphs are used in special cases described later.

The routing scheme of the network is defined in terms of the graph model. Messages are sent from source v_i to destination v_j along an (i,j)-path. A sequence $p = v_1 v_2 \dots v_n$ is a path if there is an edge between node v_i and node v_{i+1} and if $v_i \neq v_j$ for all i,j. (Note that a path is typically defined as an alternating sequence of vertices and edges in which no vertex is repeated. Since only simple graphs are considered, a path can be defined by a sequence of vertices.) Also, p is called an (i,j)-path if $v_1 = v_i$ and $v_n = v_j$; the path is denoted by $p(i,j)$. Direct communication implies that the path $p(i,j)$ used to send a message from node v_i to node v_j has length one, i.e., $p(i,j)$ is precisely the edge $e = v_i v_j$. In indirect communication, an (i,j)-path uses intermediate nodes to forward a message to its destination, i.e., $p(i,j) = v_i \dots v_k \dots v_j$, where v_k is an intermediate node.

To initiate communication between nodes in a direct network, a message-passing request is made. A message-passing request is an ordered pair $M = (S,D)$, where $S \subseteq V(G)$ is the source set and $D \subseteq V(G)$ is the destination set. Message passing paradigms are classified according to the sizes of the source and destination sets. The multicast communication paradigm that was described above has the characteristics that $|S| = 1$ and $|D| \geq 1$. In unicast, $|D| = 1$ and in broadcast, $D = V(G) - S$.

Communication Models

One constraint that a UBM algorithm must consider is the communication model that applies to the direct network. Three communication models are defined by Farley in [14]. The distinguishing characteristic in the three variations is the requirements on the paths that can be used during simultaneous message transmissions. In the neighbor-switching model, all communication is direct, i.e., each path used consists of a single edge. In the line-switching model, all the paths used simultaneously must be edge disjoint. In the node-switching model, all the paths used simultaneously must be node disjoint.

As the names suggest, these communication models depend on the type of switching used by the direct network. Store-and-forward switching is best described by the neighbor-switching model. In the neighbor-switching model, a message can only be sent to a neighboring node and the time needed to send a message to a neighboring node is considered to be one unit. The line-switching model is applicable to current wormhole-routed commercial multicomputers with generic nodes as depicted in Figure 1. Any call requires unit time because message transmission is relatively independent of path length. In addition, each router can relay several messages simultaneously provided the messages do not require the same external channels (i.e., paths are edge disjoint). Thus, two calls that use the same node can be made simultaneously, as long as the external channels used in each path are disjoint. Because most current commercial multicomputers use cut-through techniques, here focus will be on

the line-switching model of communication. Node-switching does not have an analogous switching technique; however, notice that the node-switching criterion also satisfies the line-switching criterion.

Calling Schedules

The typical implementation for multicast is to use a calling schedule, first introduced by Farley [14,15]. (Farley assumes a graph model. All concepts are easily extended to a digraph model by making edges directed.) A calling schedule consists of several time steps of unit length during which one or more unicasts can be executed in parallel. Each unicast is characterized by three parameters: source, destination and time step. In order to clearly indicate the path used to perform a unicast, McKinley et al. [8] introduce a fourth parameter so that a unicast call is represented as an ordered quadruple $(i, j, p(i, j), t)$ which is read as "Node i sends a message to node j along path $p(i, j)$ during time step t." For a multicast request $M = (S, D)$, where $S = \{s\}$ and $D = \{d_1, d_2, \ldots, d_n\}$, the informed set of M at time step t with respect to a calling schedule C, denoted $I_t^C(M)$, is defined to be the set of all nodes from $S \cup D$ that have received the message after time step t is executed under calling schedule C [16]. Thus, $I_0^C(M) = \{s\}$ for any C and any M.

There are various design principles to consider when creating a calling schedule for a given multicast request. First, a UBM algorithm must create a calling schedule that adheres to the constraints of the system. For example, the algorithm can only choose paths from the routing scheme and it must schedule calls that are permissible by the switching mechanism. If the system has a one-port architecture, then a node can only inform one destination in a single time step. In addition, a UBM implementation should not involve nodes that are not included in the multicast request. Delivery of a message to a nondestination node interrupts useful work and is therefore prohibited. Another desirable characteristic is that each destination receives the messages exactly once. Clearly there is no advantage to delivering a message more than once to any destination. Furthermore, this would require some mechanism for determining that a message is a duplicate. Thus, generality is not lost by restricting the attention to algorithms that deliver a message exactly once to each destination. Finally, only previously informed nodes can be the source of a multicast and the UBM algorithm must inform all destinations in a finite amount of time. A UBM algorithm for multicast request $M = (s, D)$ that meets these design requirements is called a legal calling schedule for M. Below, the formal requirements are presented for a legal calling schedule C that implements a multicast request M under the line-switching model. (Farley's requirements for a corresponding legal line-switching calling schedule include only Conditions ii and vi.) The requirements for legal calling schedules that implement node- and neighbor-switching are identical except for an appropriate modification to Condition vi.

i. If $(i, j, p(i, j), t) \in C$ and $(m, n, p(m, n), t) \in C$ then $i \neq m$ and $j \neq n$.

ii. For all $(i, j, p(i, j), t) \in C$, $i \in I_{t-1}^C(M)$.

iii. For all $(i, j, p(i, j), t) \in C$, $j \notin I_{t-1}^C(M)$.

iv. There exists a time step t such that $I_t^C(M) = S \cup D$ and t is called the length of C.

v. For all $(i, j, p(i, j), t) \in C$, $i, j \in S \cup D$.

vi. If $(i, j, p(i, j), t) \in C$ and $(m, n, p(m, n), t) \in C$ are distinct unicasts, then $p(i, j)$ and $p(m, n)$ are edge-disjoint.

Performance Measures

Once the constraints of the system have been determined, there may be several legal calling schedules that satisfy the multicast request. Thus, it is necessary to evaluate the performance of a UBM algorithm so that the most efficient calling schedule can be selected.

In general, efficient algorithms should use as little of the system's resources as possible. Typically two resources have been considered,

namely, time and traffic. Because communication overhead is so costly, it is desirable to minimize the amount of time (or number of steps) necessary to complete a multicast request. More formally, one objective is to find an algorithm that always produces legal calling schedules of minimum length. A time efficient UBM algorithm is one that minimizes the number of steps needed to perform the multicast.

Another way to measure performance is by the number of communication links used. Using a small number of communication lines keeps the traffic low and allows better overall system performance. Thus, another objective is to minimize the sum of the lengths of the paths used in a legal calling schedule. A UBM algorithm that minimizes the number of communication lines used is traffic efficient.

Variations

The majority of the work on multicast has been done on direct networks with the following properties:

i. Each node of a direct network can only send and receive a single message during a time step (i.e., the network has a one-port architecture).

ii. Multicast is implemented by a calling schedule, in which several unicast calls can be made during a single step that requires unit time.

iii. The communications network is fault free; that is, if the unicast call $(i, j, p(i, j), t)$ is made, the message will be delivered reliably along the path that is specified.

Several variations of these assumptions have been studied. One variation is to assume that the direct network is based on a k-port architecture; that is, each node can send and receive k messages in a single time step [17–20]. Multicast has also been studied for multicomputer systems that are not based on direct networks. For example, multicast has been studied in indirect networks [21] and ATM networks [22]. Another variation involves multicast implementations that are not based on unicast; the two approaches are tree-based and path-based multicasts [4,23]. In these approaches, the destinations are specified in the message header. In a tree-based approach, a spanning tree rooted at the source node is found and the multicast message is propagated along the tree. In particular, the message follows a common path until a branching point is encountered; the message is then replicated and sent along each branch, where the message header contains only the destinations of the branch to which it is sent. In path-based multicast, the message is sent along a path (or multiple paths) that goes through each destination node. If the router discovers that the message is destined for its associated processor, it both copies the incoming message to the local processor and forwards the message along the path. No branching occurs in this approach. Variations on the model for message transmission have also been studied. Multicast algorithms have been developed under the Log P model [24], Postal model [25] and Parameterized model [26]. These algorithms can be used in indirect networks as well as direct networks. Other considerations that have not been discussed here are the issues of deadlock and fault tolerance. Several of these variations will be briefly discussed as they arise in later sections.

NEIGHBOR-SWITCHING

Recall that in the neighbor-switching communication model, a node may only call one of its neighbors. For example, node $(1, 1)$ in Figure 3 can only make calls to nodes $(1, 2)$ and $(2, 1)$. In this model, it is assumed that the cost to deliver a message is uniform for all communication links, i.e., a message can be delivered in unit time.

Before circuit routing became popular, most networks used store-and-forward routing. As mentioned previously, networks that use store-and-forward routing are best modeled by the neighbor-switching assumption. Consequently, much of the early work on multicast in networks assumed the neighbor-switching

communication model. Notice that the concept of free routing vs. oblivious routing is not applicable in the neighbor-switching model. There is only one "path" (of length one) between a node v and its neighbor w. Thus, node v is both free to use any available path and restricted to a specific path when sending a message to w.

One drawback of the neighbor-switching model is that not all subsets of nodes may be reached using only local calls. For example, suppose a multicast request with $S = \{(3,2)\}$ and $D = \{(1,2),(1,4),(2,2),(3,1),(3,3),(4,3),(4,4)\}$ is made in the network shown in Figure 3. Under the neighbor-switching assumption, there is no way to deliver the message to node $(1,4)$ using only nodes in $S \cup D$. Thus, it is not surprising that most of the work that uses the neighbor-switching model concentrates on the broadcast problem, where $S \cup D = V(G)$. Other authors have allowed vertices which are not source or destination to be involved, so that multicast can be addressed [27–29]. In this case, nodes that do not need the message are interrupted and required to forward the message.

In the context of neighbor-switching, two classes of problems have been considered. The first problem is to find an efficient way to multicast (or broadcast) in a given graph [27,30,31]. The second problem is to construct graphs that will always be able to do multicast (or broadcast) efficiently [15,32–39].

Finding Efficient UBM Multicast Algorithms

In general, a graph is given and it is desirable to multicast from a given source node. A calling schedule (see the previous sections) is used to implement the multicast. The source node makes a call to one of its neighbors and the other nodes may also make calls after they have received the message. This continues until all destination nodes are informed. Typically, the edges that the messages travel along make up a multicast tree. Again, because some nodes are unreachable using only neighbors and broadcast is more widely addressed within this context. Figure 5 shows two broadcast trees for K_8, the complete graph on eight vertices, where the source node is node 1. The number on each edge shows the time step during which the edge is used to deliver a message. The broadcast tree in (b) requires seven steps while the tree in (c) requires only three. Many broadcast trees exist for a single graph. One problem that has been considered is how to choose the "best" broadcast tree.

Slater et al. show that the problem of determining the minimum amount of time required to broadcast from an arbitrary vertex of an arbitrary graph is NP-complete [30]. Thus, Slater et al. focus on broadcasting in trees. In particular, they develop an algorithm for finding the broadcast center of a tree. In general, a center of a graph is defined as follows:

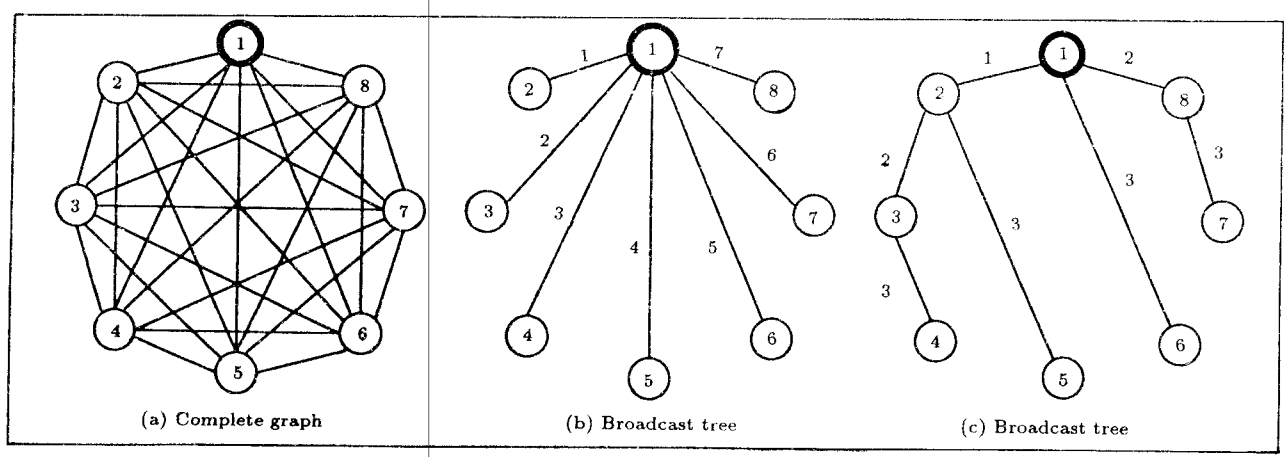

Figure 5. Broadcast trees for K_8.

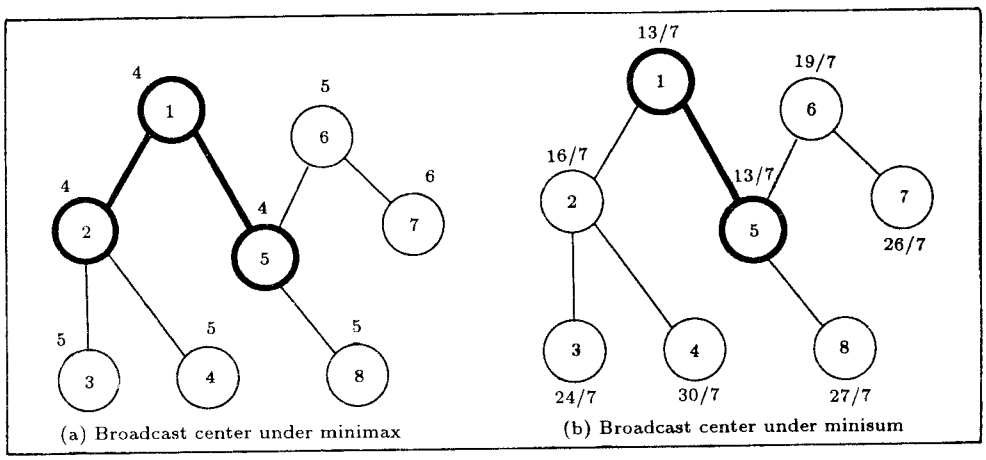

Figure 6. Broadcast center of a tree.

(1) each vertex is assigned a value according to some measure, e.g., weight, distance, or time; (2) the center is the set of vertices that have minimum value. In [30], the measure being minimized is the number of time steps needed to broadcast from a vertex. That is, the broadcast center of a tree is the set of vertices in the tree from which broadcast can be completed in the least amount of time. Slater et al. show that the broadcast center of a tree is always a star with two or more vertices. For example, consider the tree in Figure 6a. The number shown beside each node is the broadcast time from that node. Thus, the subgraph shown in bold lines is the broadcast center of the tree.

Koh and Tcha [31] use a different measure to find the broadcast center of a tree. They minimize the average time at which a vertex in the tree receives the message; this is called the minisum criterion, whereas the function used by Slater et al. is called the minimax criterion. The average time that each node receives the broadcast message is shown in Figure 6b and the center under the minisum criterion is again shown in bold. The center is not the same as that for the minimax criterion, however, this is not always the case.

Bharath-Kumar and Jaffe [27] consider multicast in arbitrary networks, and they focus on traffic efficiency. As mentioned earlier, not all subsets of nodes can be reached using local calls. Thus, some of the nodes used to deliver the message may not be destination nodes. Furthermore, the authors permit any subgraph to be used to deliver messages. Suppose a node must deliver a message to a non-neighbor. A common technique is for the node to use local information to determine which link it will use to forward the message and therefore which node is the next to be responsible for delivering the message. Because the routing is done in this distributed manner, any subgraph may be used for the multicast. For example, Figure 7 shows a graph that is used for multicast that is not a tree. Node 1 is the originator of the message and it is responsible for sending messages to nodes 6 and 7. Node 1 chooses to send the message destined for node 6 through intermediate node 2 and to send the message destined for 7 through intermediate node 5. Node 2 sends the message directly to node 6. Node 5 chooses to send the message destined for 7 through node 6, thus creating a cycle in the multicast graph.

Within this context, Bharath-Kumar and Jaffe define two criteria used to measure the efficiency of the multicast implementation. Network Cost (NC) measures the number of communication links used to deliver the message and Destination Cost (DC) measures the average delay experienced by each destination. Their goal is to minimize NC because applications like file transfer do not rely heavily on message delay. The authors explain that, while optimizing DC is relatively easy, optimizing NC is an NP-complete problem. They discuss the

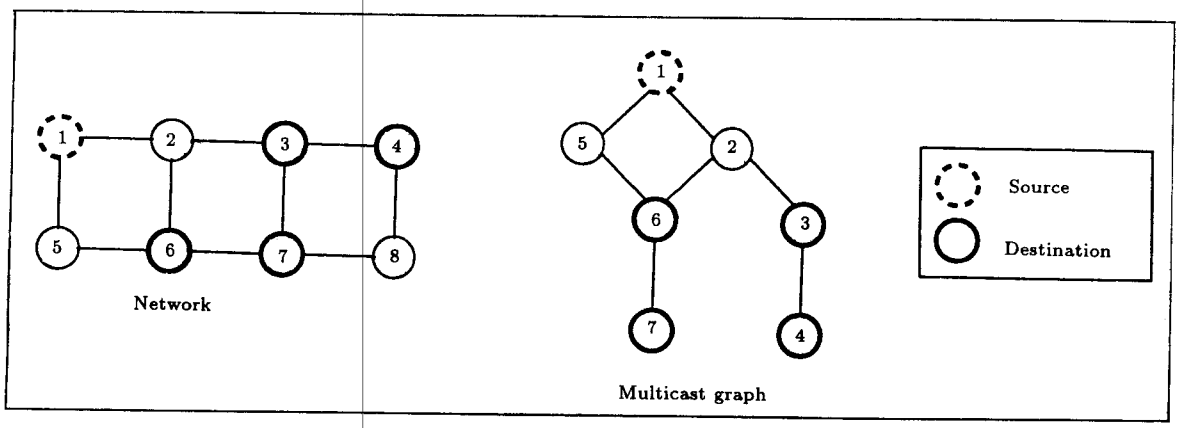

Figure 7. Multicast subgraph.

feasibility of using DC optimal algorithms to approximate NC optimal solutions and they devise several heuristics for minimizing NC.

While it may be necessary to involve non-destination nodes in a multicast tree, it is desirable to use as few additional nodes as possible since useful work on the additional nodes is interrupted to receive messages that they do not need. Choi, Esfahanian and Ni proposed the Optimal Multicast Tree (OMT) problem [28] in which a distance preserving multicast tree T with as few additional nodes as possible must be found. They show that OMT is NP-complete for arbitrary graphs. Lan, Esfahanian and Ni consider multicast in hypercube topologies and develop heuristic algorithms for finding an OMT [29].

Variations in Finding Multicast Algorithms

Finding efficient neighbor-switching UBM algorithms has also been studied in several variations on the system described in the first section. Choi and Esfahanian [17] consider the problem of multicasting in a network where a node can send a message to more than one neighbor simultaneously; that is, they do not assume a one-port architecture. They model the network as a graph and consider both time and traffic efficiency. In particular, they propose the Optimal Communication Tree (OCT) problem, in which finding a multicast (or broadcast) tree T that preserves the distances in G from each node to the source and has the fewest number of nodes is desirable. In general, OCT is NP-complete. Choi and Esfahanian show that the OCT problem is NP-complete even for the n-cube and for graphs whose maximum degree is at most three. They also present some heuristics for the OCT problem.

Lin and Ni describe a few multicast techniques that are not unicast-based [40]. They discuss the multicast path model in which a path that passes through all destinations is used to deliver the message. In this model, the message does not need to be replicated. Within this context, they define the Optimal Multicast Path (OMP) problem, which is to construct a multicast path with shortest length in a graph G. Lin and Ni also describe the Multicast Cycle (MC) problem, in which a cycle that contains the multicast destinations must be constructed. This is used to avoid having to send separate acknowledgment from each destination to the source. Again, they define the Optimal Multicast Cycle (OMC) problem, which is to find the shortest multicast cycle. They show that both OMP and OMC are NP-complete and discuss heuristics for both problems. They also define the Optimal Multicast Tree (OMT) problem, which is identical to the OCT problem described above.

Gaber and Mansour study broadcast in radio networks [41]. In this problem, if two stations neighboring a node transmit simultaneously, their messages will collide and neither of the messages will be received at the node. Thus, a node can receive a message only if

exactly one of its neighbors transmits during a time slot. Gaber and Mansour develop a broadcast algorithm that requires $\Omega(D)$ time slots for sufficiently large D, where D is the diameter of the graph.

Efficient Broadcast Graphs

Other work has focused on characterizing the types of graphs in which broadcast can always be done efficiently. The broadcast time from a vertex v in a graph G is the minimum number of time steps needed to broadcast a message from v. The broadcast time of a graph G, $b(G)$, is the maximum broadcast time from any vertex in G. Because the number of informed nodes can at most double in each step, $b(K_n) = \lceil \lg n \rceil$. (The convention $\lg n = \log_2 n$ is used.) However, some edges can be removed from K_n and the resulting graph still has broadcast time $\lceil \lg n \rceil$. A minimal broadcast graph on n vertices is a graph G with the properties that $b(G) = \lceil \lg n \rceil$, however, any subgraph G' of G has $b(G') > \lceil \lg n \rceil$. Furthermore, $B(n)$ is the minimum number of edges in any minimal broadcast graph and a Minimum Broadcast Graph (MBG) on n vertices is a minimal broadcast graph with $B(n)$ edges. An MBG provides a network topology in which broadcast can be completed from every node in a minimum number of time steps. A substantial amount of work has been done in this area [15,32–39,42]. Fraigniaud and Lazard provide an extensive survey of this work in [43].

LINE-SWITCHING

It was explained previously that current wormhole-routed direct networks, in which each node has a processor and separate router, allow "long distance" calls to non-neighboring nodes to be made in unit time. Unlike the neighbor-switching model, several physical paths may exist between a source node s and destination node d. If the system allows any (s, d)-path to be used, it is said to employ a free routing scheme. If the system specifies a unique (s, d)-path for sending a message from s to d, the system is said to use oblivious routing.

Free Routing

Farley first addressed line-broadcasting (broadcasting in a general graph under the line-switching assumption) [14]. He shows that under the line-switching assumption, broadcasting can be completed in minimum time ($\lceil \lg n \rceil$ time units) in any connected network of n nodes that employs free routing, regardless of message originator.

McKinley, et al. addressed the problem of performing multicast in wormhole-routed direct networks [8]. They showed that there exists a lower bound implementation, i.e., it requires exactly $\lceil \lg(d+1) \rceil$ steps to deliver the message to d destinations, for any multicast request in systems that admit free unicast communication, regardless of topology. The general idea of their algorithm is as follows. First, construct a trail (a trail is a sequence of vertices and edges in which no edge is repeated) that includes the source node and all the destinations nodes. The message is delivered from the source to a "middle" destination and the trail is then broken into two equal length subtrails each of which contains an informed node. This is continued recursively until all destinations are informed, thus $\lceil \lg(d+1) \rceil$ steps are required, where d is the number of destinations. Neither Farley nor McKinley et al. consider traffic efficiency.

Kane and Peters [44] consider time and traffic efficient broadcasting in cycles. They construct algorithms that always use a minimum number of phases ($\lceil \lg n \rceil$) and that also minimize the number of communication links used in each phase.

Oblivious Routing

With respect to time efficiency, the problem of multicasting in systems that permit free routing is solved, i.e., optimal algorithms have been found. Unfortunately, most commercial machines offer only oblivious unicast routing, particularly in wormhole-routed networks where deadlock would result if free routing were used. Consequently, it is important to study efficient UBM algorithms for systems that employ oblivious routing.

McKinley et al. [8] use the calling schedule

implementation described previously to develop lower-bound (on time) algorithms for doing multicast in hypercubes and meshes. Their algorithm was based on the algorithm described above in which a trail containing the source and all destinations is recursively divided into equal subtrails. Their algorithm was later extended to do multicast in torus networks [45].

In [8,45] sending a message from one node to another is considered to require unit time regardless of the message size or path length. Fraigniaud and Peters [46] describe a method for measuring communication time that does rely on message size and path length. In particular, the time to send a message of length ℓ over a path of length d is $\alpha + d\delta + \ell\tau$, where α represents startup time, δ represents the switching delay and τ is the propagation time. They propose a circuit-switched algorithm for broadcast in a torus that requires time $d\lg(n)\alpha + d\frac{n}{2}\delta + d\lg(n)\ell\tau$.

Variations

Variations of this problem have also been studied. McKinley and Trefftz study broadcast in all-port wormhole-routed hypercubes [18]. All-port architectures have the same number of internal and external channels (see Figure 1). They develop the Double Tree (DT) algorithm and show that it completes broadcast in $\lceil \frac{n}{2} \rceil$ steps in an n-cube.

Tsai and McKinley also study all-port architectures. They introduce the Extended Dominating Node model for collective communication. A Dominating Set in a graph G is a set D of nodes with the property that all nodes in G are in D or adjacent to one or more nodes in D. An Extended dominating set in a graph is a set E with the property that all nodes not in E have edge disjoint paths to nodes in E. To accomplish multicast, a message is sent to the nodes in the extended dominating set in one or more time sets. Because of the property of the dominating set, the multicast can be completed in one last step by sending the message along the disjoint paths to the remaining destinations. Tsai and McKinley present algorithms for broadcast in meshes [47,48] and tori [20] and they give a generalized method for multicast and other collective communication operations in [19].

The work that has been discussed so far has assumed that a message is delivered in a single step, or in "unit" time. While this is a fairly good model, other frameworks have been developed to more closely model communication on actual machines. Such models include the Postal model, the Log P model and the Parameterized model. In the Postal model, a message transmission requires λ units. Thus, a message sent at time t will arrive at its destination at time $t + \lambda - 1$. In [25], a method is given for constructing optimal multicast trees under this model. The authors assume a logical complete-graph topology, i.e., they assume that communication is uniform between any pair of nodes in the network. The Log P model is a generalization of the Postal model. In the Log P model, L is the latency (time to send a message), o is the overhead (time that a processor is involved in transmitting or receiving and cannot perform other operations) and g is the gap (time between successive message transmissions). Karp et al. discuss optimal broadcast under this model in [24]. Although the Log P model is more general than the Postal model, it also has shortcomings in modeling communication on real machines. Park, et al. introduce the Parameterized model to more closely represent actual communication. In this model, message transmission is comprised of three parameters: the sending latency (t_{send}), the receiving latency (t_{recv}) and the network latency (t_{net}). Unfortunately, each of these parameters is rather difficult and costly to measure. Thus, the easily measured parameters t_{end}, or end-to-end latency, and t_{hold}, holding time, are introduced. As in [25], Park, et al. assume a logical complete-graph topology. They describe methods for constructing optimal multicast trees for one-port and multiport architectures under the Parameterized model.

As mentioned before, multicast has also been studied for parallel systems that are not based on direct networks. Multicast communication in ATM networks used for parallel

computing is studied in [22]. The authors show that multicast trees can provide a substantial benefit over a separate addressing technique, in which the source sends the message sequentially to each destination. Xu and Ni address multicast communication in multistage networks that use turnaround routing (e.g., TMC CM-5, Meiko CS-2 and IBM SP-2) [21]. They develop the U-min algorithm for optimal multicast in such networks. This algorithm is based on the algorithms presented in [8,45] for optimal multicast in meshes, hypercubes and tori.

NODE-SWITCHING

The node-switching model has not been studied as extensively as the other models, since it does not have a corresponding switching technique in direct networks.

Free Routing

In [14], Farley shows that it may be impossible to find a minimum length, i.e., $\lceil \lg n \rceil$, node-disjoint broadcast schedule in an arbitrary tree with n nodes. However, he also shows that for all n there does exist a tree with n nodes that allows minimum time node-disjoint broadcasting to be done.

Oblivious Routing

Birchler, Esfahanian and Torng use node-disjoint multicast to find a heuristic line-switching UBM algorithm for arbitrary topologies [16,49]. They define a realistic class of oblivious routing schemes called source limited inclusive routing, which includes both xy-routing on a mesh and e-cube routing on a hypercube. They study multicast within this context. In particular, source limited inclusive routing schemes are modeled by directed trees. Given a directed tree T rooted at r, they want to send a message from r to a subset of the nodes in T. They show that in a directed tree, the number of time steps for a node-disjoint multicast schedule is at most twice the number of steps for an edge-disjoint multicast schedule. In addition, they show that, under the node-disjoint model, a multicast problem in a directed tree T can be transformed to an equivalent broadcast problem in a related ditree T'. In [49], they present a dynamic programming algorithm that computes a minimum length node-disjoint broadcast schedule for any directed tree T. Consequently, their algorithm is an order two approximation for performing multicast under the line-switching assumption.

OPEN ISSUES

Various components of the multicast problem: network model, communication models, routing schemes and performance measures have been discussed. The bulk of the work has been done using a graph model for the network. This necessarily implies that the network is symmetric, i.e., the (x, y)- and (y, x)-paths use the same edges. Very little has been done in directed graphs, which model non-symmetric networks. Consequently, the results cannot generally be applied to direct networks that are not symmetric. To address non-symmetric communication, a more general model is needed.

Both the neighbor-switching and line-switching communication models have been studied. Finding time efficient and traffic efficient multicast algorithms in systems that use neighbor-switching is NP-complete. Heuristics for these problems have been developed, however no tight bounds on the heuristics have been given. Another open area is in the development of minimum broadcast graphs.

Recall that direct networks that use circuit routing techniques, such as wormhole routing, conform to the line-switching communication model. Thus, multicast under this model is more applicable to current multiprocessor systems. Because several paths may exist between a source and destination pair, many authors have considered free routing. Optimal-time multicast algorithms have been developed for arbitrary topologies that admit free routing. Unfortunately, most commercial multicomputers use some type of oblivious routing. Optimal time UBM algorithms have been developed for hypercubes, meshes and tori. However, none of the results for oblivious routing are general, i.e.,

they do not apply to any topology that uses an arbitrary oblivious routing scheme.

Finally, both time and traffic have been used to measure the performance of multicast algorithms. However, traffic efficient UBM algorithms have not been addressed for arbitrary systems that admit free routing or oblivious routing.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their detailed and insightful comments. This work was supported in part by NSF grant MIP-9204066.

REFERENCES

1. Hwang, K. *Advanced Computer Architecture: Parallelism, Scalability, Programmability*, McGraw-Hill Series in Computer Science, New York, McGraw-Hill, Inc., USA (1993).
2. DeMara, R.F. and Moldovan, D.I. "Performance indices for parallel marker-propagation", in *Proceedings of the 1991 International Conference on Parallel Processing* (St. Charles, Illinois), pp 658–659 (Aug. 1991).
3. Kumar, V. and Singh, V. "Scalability of parallel algorithms for the all-pairs shortest path problem", Tech. Rep. ACT-OODS-058-90, Rev. 1, MCC (Jan. 1991).
4. Lin, X., McKinley, P.K. and Ni, L.M. "Deadlock-free multicast wormhole routing in 2-D mesh multicomputers", *IEEE Transactions on Parallel and Distributed Systems*, **5**, pp 793–804 (Aug. 1994).
5. Lenoski, D., Laudon, J., Gharachorloo, K., Gupta, A. and Hennessy, J. "The directory-based cache coherence protocol for the dash multi-processor", in *Proceedings of the 17th Annual International Symposium on Computer Architecture*, pp 148–159 (May 1990).
6. "Message passing interface forum", Tech. Rep. CS-93-214, University of Tennessee (Nov. 1993).
7. Ni, L.M. and McKinley, P.K. "A survey of wormhole routing techniques in direct networks", *IEEE Computer*, **26**, pp 62–76 (Feb. 1993).
8. McKinley, P.K., Xu, H., Esfahanian, A.-H. and Ni, L.M. "Unicast-based multicast communication in wormhole-routed networks", *IEEE Transactions on Parallel and Distributed Systems*, **5**, pp 1252–1265 (Dec. 1994).
9. Leighton, F.T. *Introduction to Parallel Algorithms and Architectures*, Morgan Kaufmann Publishers, San Mateo, CA, USA (1992).
10. Intel, Supercomputer Systems Division, Intel Corporation, Beaverton, OR 97006, *Paragon XP/S Product Overview* (1991).
11. NCUBE Company, *NCUBE 6400 Processor Manual* (1990).
12. Felperin, S.A., Gravano, L., Pifarre, G.D. and Sanz, J.L.C. "Routing techniques for massively parallel communications", *Proc. IEEE*, **79**, pp 488–503 (April 1991).
13. Stunkel, C.B., Shea, D.G., Abali, B., Atkins, M., Bender, C.A., Grice, D.G., Hochschild, P.H., Joseph, D.J., Nathanson, B.J., Swetz, R.A., Stucke, R.F., Tsao, M. and Varker, P.R. "SP2 high-performance switch", *IBM Systems Journal*, **34**(5), pp 185–204 (1995).
14. Farley, A.M. "Minimum-time line broadcast networks", *Networks*, **10**, pp 59–70 (1980).
15. Farley, A.M. "Minimal broadcast networks", *Networks*, **9**, pp 313–332 (1979).
16. Birchler, B.D., Esfahanian, A.-H. and Torng, E. "Toward a general theory of unicast-based multicast communication", in *Lecture Notes in Computer Science*, M. Nagl, Ed., **1017**, pp 237–251, Springer-Verlag (1995).
17. Choi, H.-A. and Esfahanian, A.-H. "A message-routing strategy for multicomputer systems", *Networks*, **22**, pp 627–646 (1992).
18. McKinley, P.K. and Trefftz, C. "Efficient broadcast in all-port wormhole-routed hypercubes", in *Proceedings of 1993 International Conference on Parallel Processing*, **2** (St. Charles, Illinois), pp 288–291 (Aug. 1993).
19. Tsai, Y. and McKinley, P.K. "An extended dominating node approach to collective communication in wormhole-routed networks", *IEEE Transactions on Parallel and Distributed Systems* (1996).
20. Tsai, Y. and McKinley, P.K. "A broadcast algorithm for all-port wormhole routed torus networks", *IEEE Transactions on Parallel and*

Distributed Systems, **7**, pp 876–885 (Aug. 1996).

21. Xu, H., Gui, Y.-D. and Ni, L.M. "Optimal software multicast in wormhole-routed multistage networks", in *Proceedings of the 1994 Supercomputing Conference*, pp 703–712 (Nov. 1994).
22. Huang, C. and McKinley, P.K. "Communication issues in parallel computing across ATM networks", *IEEE Parallel and Distributed Technology*, pp 73–86 (Winter 1994).
23. Tseng, Y.-C., Panda, D.K. and Lai, T.-H. "A trip-based multicasting model in wormhole-routed networks with virtual channels", *IEEE Transactions on Parallel and Distributed Systems*, **7**, pp 138–150 (Feb. 1996).
24. Karp, R., Sahay, A., Santos, E. and Schauser, K. "Optimal broadcast and summation in the Log P model", in *Fifth Symposium on Parallel Algorithms and Architectures* (Jun. 1993).
25. Bruck, J., Coster, L.D., Dewulf, N., Ho, C.-T. and Lauwereins, R. "On the design and implementation of broadcast and global combine operations using the postal model", *IEEE Transactions on Parallel and Distributed Systems*, **7**, pp 256–265 (March 1996).
26. Park, J.-Y.L., Choi, H.-A., Nupairoj, N. and Ni, L.M. "Construction of optimal multicast trees based on the parameterized communication model", in *1996 International Conference on Parallel Processing* (1996).
27. Bharath-Kumar, K. and Jaffe, J.M. "Routing to multiple destinations in computer networks", *IEEE Transactions on Communications*, **31**, pp 343–351 (March 1983).
28. Choi, Y., Esfahanian, A.-H. and Ni, L.M. "One-to-k communication in distributed-memory multiprocessors", in *Proceedings 25th Annual Allerton Conference on Communication, Control and Computing*, pp 268–270 (Sept. 1987).
29. Lan, Y., Esfahanian, A.-H. and Ni, L.M. "Multicast in hypercube multiprocessors", *J. of Parallel and Distributed Computing*, **8**, pp 30–41 (1990).
30. Slater, P.J., Cockayne, E.J. and Hedetniemi, S.T. "Information dissemination in trees", *Society for Industrial and Applied Mathematics*, **10**, pp 692–701 (Nov. 1981).
31. Koh, M.J. and Tcha, W.D. "Local broadcasting in a tree under minisum criterion", *Networks*, **23**, pp 71–79 (1993).
32. Mitchell, S. and Hedetniemi, S. "A census of minimum broadcast graphs", *J. Combin. Inform. Systems Sci.*, **5**, pp 141–151 (1980).
33. Bermond, J.-C., Hell, P., Liestman, A. and Peters, J. "Sparse broadcast graphs", *Discrete Applied Mathematics*, **36**, pp 97–130 (1992).
34. Farley, A., Hedetniemi, S., Mitchell, S. and Proskurowski, A. "Minimum broadcast graphs", *Discrete Mathematics*, **25**, pp 189–193 (1979).
35. Grigni, M. and Peleg, D. "Tight bounds on minimum broadcast networks", *SIAM J. Discrete Math.*, **4**, pp 207–222 (1991).
36. Liestman, A.L. and Peters, J.G. "Broadcast networks of bounded degree", *SIAM J. Disc. Math.*, **1**, pp 531–540 (Nov. 1988).
37. Bermond, J.-C., Hell, P., Liestman, A. and Peters, J. "Broadcasting in bounded degree graphs", *SIAM J. Disc. Math.*, **5**, pp 10–24 (Feb. 1992).
38. Liestman, A.L. and Peters, J.G. "Minimum broadcast digraphs", *Discrete Applied Mathematics*, **37/38**, pp 401–419 (1992).
39. Gargano, L., Liestman, A., Peters, J. and Richards, D. "Reliable broadcasting", *Discrete Applied Mathematics*, **53**, pp 135–148 (1994).
40. Lin, X. and Ni, L.M. "Multicast communication in multicomputer networks", *IEEE Transactions on Parallel and Distributed Systems*, pp 1105–1117 (Oct. 1993).
41. Gaber, I. and Mansour, Y. "Broadcast in radio networks", in *Proceedings 6th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp 577–585 (1995).
42. Chau, S. and Liestman, A. "Constructing minimal broadcast networks", *J. Combin. Inform. System. Sci.*, **10**, pp 110–122 (1985).
43. Fraigniaud, P. and Lazard, E. "Methods and problems of communication in usual networks", *Discrete and Applied Mathematics*, pp 79–133 (Sept. 1994).
44. Kane, J.O. and Peters, J.G. "Line broadcasting in cycles", Tech. Rep. CMPT TR 94-11, School of Computing Sciences, Simon Fraser University, Burnaby, B.C., Canada (Dec. 1994).

45. Robinson, D.F., McKinley, P.K. and Cheng, B.H. "Optimal multicast communication in wormhole-routed torus networks", in *1994 International Conference on Parallel Processing*, pp I–134–I–141 (1994).

46. Fraingiaud, P. and Peters, J. "Structured communication in torus networks", in *Hawaii Int. Conf. on Systems Science*, IEEE (Jan. 1995).

47. Tsai, Y. and McKinley, P.K. "Broadcast in all-port wormhole-routed 3D mesh networks using extended dominating sets", in *International Conference on Parallel and Distributed Systems*, pp 120–127 (Dec. 1994).

48. Tsai, Y. and McKinley, P.K. "A dominating set model for broadcast in all-port wormhole-routed 2D mesh networks", in *Proceedings of the 8th ACM International Conference on Supercomputing*, pp 126–135 (July 1994).

49. Birchler, B.D., Esfahanian, A.-H. and Torng, E. "Information dissemination in restricted routing networks", in *The International Symposium on Combinatorics and Applications*, Tianjin, China, pp 33–43 (June 1996).