Tree-Based Multicasting in Wormhole-Routed Irregular Topologies*

Ran Libeskind-Hadas Dominic Mazzoni Ranjith Rajagopalan
Department of Computer Science
Harvey Mudd College
Claremont, CA 91711

Abstract

A deadlock-free tree-based multicast routing algorithm is presented for all direct networks, regardless of interconnection topology. The algorithm delivers a message to any number of destinations using only a single startup phase. In contrast to existing tree-based schemes, this algorithm applies to all interconnection topologies, requires only fixed-sized input buffers that are independent of maximum message length, and uses a simple asynchronous flit replication mechanism. The theoretical basis of the technique used here is sufficiently general to develop other tree-based multicasting algorithms for regular and irregular topologies. Simulation results demonstrate that this tree-based algorithm provides a very promising means of achieving very low latency multicast.

1 Introduction

Recently, networks of workstations (NOWs) have emerged as an inexpensive alternative to massively parallel multicomputers. These networks generally comprise a collection of routing switches and workstations interconnected in an irregular topology. In order to minimize network latency and achieve high-bandwidth communication, recent experimental and commercial switches for NOWs implement wormhole routing [2, 13]. Although deadlock-free wormhole routing algorithms have been proposed for a number of regular topologies such as meshes, tori, and hypercubes, the design of deadlock-free routing algorithms in irregular topologies introduces new challenges.

Communication in the network can be either unicast (one-to-one) or multicast (one-to-many). Efficient multicast communication is essential in a wide variety of applications as well as in a number of fundamental operations such as barrier synchronization [17], cache coherency in distributed shared-memory architectures [5], and clock synchronization [1], among others.

Several unicast-based schemes have been proposed to support multicast communication in irregular topologies. These strategies use a deadlock-free unicast algorithm to distribute a multicast message to its destinations in several communication phases. Each communication phase incurs a startup latency that can be several orders of magnitude larger than the actual network latency. Moreover, a lower bound on the number of unicast communication phases required to distributed a message to d destinations is known to be $\lceil \log_2(d+1) \rceil$ [10]. For this reason, hardware-supported multicast solutions have recently been proposed to substantially reduce the number of communication phases required. Some algorithms use hardware-supported *path-based* techniques to deliver a message with very few worms while other techniques use *tree-based* techniques to deliver a message in a single worm. Sivaram *et al.* [14] have recently shown that tree-based techniques offer a very promising means of achieving extremely efficient multicast routing. However, existing tree-based algorithms for direct networks avoid deadlock by either requiring complex synchronization mechanisms or by requiring that intermediate routers be able to buffer the message in its entirety, thereby limiting the length of packets to be no longer than the size of these buffers.

In this paper we present a tree-based multicasting algorithm for arbitrary topologies that guarantees freedom from deadlock for arbitrarily long messages, even with single-flit input buffers and simple routing hardware. To the best of our knowledge, this is the first deadlock-free tree-based wormhole routing algorithm that provides a general solution for multicasting in any direct network.

*This work was supported in part by the National Science Foundation under grant CCR-9418311.

The remainder of this paper is organized as follows. In Section 2 we review related work on tree-based multicasting in regular and irregular networks. In Section 3 we describe our new algorithm, *Single Phase Adaptive Multicast (SPAM)*. Section 4 contains experimental results and we conclude with directions for future research in Section 5.

2 Related Work

Multicast routing algorithms can be classified as *unicast-based*, *path-based*, and *tree-based*. Unicast-based multicast algorithms require no additional hardware support but incur substantial start-up delays. Path-based multicast algorithms allow a worm to contain multiple destination addresses in its header flits and use a simple hardware mechanism to allow routers to absorb flits on internal consumption channels (to the local processor) while simultaneously forwarding copies of the flits on output channels enroute to the remaining destinations [6, 12]. In this way, a single worm can be used to deliver a message to several destinations incurring only a single startup for that worm. For example, Hamiltonian path-based routing requires only two startups to send a message to any set of destinations in a mesh [6] while other *base routing conformed path* algorithms may use more startups but frequently use shorter paths to the destinations [3, 7, 12].

Tree-based algorithms attempt to deliver the message to all destinations in a single *multi-head* worm that splits at some routers and replicates the data on multiple output ports. Two approaches have been proposed for replication of data in tree-based schemes. In *synchronous replication* [11], all branches of the multi-head worm must advance in lock-step. Thus, any branch that becomes blocked causes all other branches to stop. Moreover, this scheme requires complex signaling hardware at the routers [11]. In *asynchronous replication*, different heads of a multi-head worm can progress independently through the network and *bubble flits* [11] are inserted where necessary, obviating the need for a hardware synchronization mechanism.

The problem of finding deadlock-free tree-based wormhole routing algorithms has been widely regarded in the literature as a difficult one [4, 6, 11, 16]. Malumbres *et al.* [9] present a tree-based asynchronous algorithm based on pruning blocked branches which is effective only for short messages. Sivaram *et al.* [14] present a tree-based asynchronous algorithm for arbitrary topologies. This algorithm uses the *input buffer based replication* architecture which requires that intermediate routers be able to buffer the entire packet [15]. Wang and Blough [16] present a tree-based asynchronous algorithm based on pipelined circuit switching rather than wormhole routing. The algorithm avoids deadlock by allowing backtracking but does not guarantee delivery of all messages. In other related work, several tree-based multicasting algorithms have been proposed for specific types of multistage interconnection networks [4, 15].

In this paper we present a tree-based multicasting algorithm for any direct network. The algorithm is provably free from deadlock even with only a single flit buffer per channel, uses asynchronous replication, and requires relatively simple modifications to existing routing hardware.

3 Single Phase Adaptive Multicast (SPAM)

We begin by describing the routing algorithm for unicast messages. We then describe the generalization of this technique for multicast messages.

3.1 SPAM Unicast Routing

We represent a switch-based network as an undirected graph $G = (V, E)$ where $V = V_1 \cup V_2$. The set V_1 represents the set of switches and the set V_2 represents the set of processors (workstations). Each vertex in V_2 is connected to a single vertex in V_1, representing a bidirectional channel (a pair of unidirectional channels) between a processor and a switch. In addition, a pair of vertices in V_1 may be connected, representing a bidirectional channel between the corresponding pair of switches. If a switch has k ports, the corresponding vertex in V_1 may have degree at most k. Each vertex in V_2 has degree 1.

Next, we partition the network in a fashion similar to that used in the up*/down* routing algorithm proposed by Schroeder *et al.* [13]. An arbitrary vertex in V_1 (representing a switch) is selected as the root and a spanning tree is found with respect to the root. Observe that all vertices in V_2 (representing processors) are leaves of this tree since they have degree 1. For each channel in the tree, the unidirectional component directed towards the root is defined to be in the "up" subnetwork whereas the unidirectional channel directed away from the root is defined to be in the "down" subnetwork. The cross channels (non-tree channels), which can only exist between vertices in V_1, are categorized similarly: If a cross channel goes from a lower level of the tree to a higher level of the tree then it is an up channel and if the channel goes from a higher level to a lower level then it is a down channel. If a channel goes from node u to v, both of which are at the same level in the tree, then the channel is an up channel if the ID of u is larger than the ID of v and a down channel otherwise.

In contrast to the original up*/down* unicast routing algorithm, our multicast routing algorithm distinguishes between down tree channels and down cross channels. No distinction is made, however, between up tree channels and up cross channels. A unicast message under SPAM is routed as follows: The worm uses one or more channels in the up subnetwork, followed by zero or more down cross channels, followed by one or more down tree channels. A worm must use channels in this order; once it has used a down cross channel it cannot use an up channel, and once it has used a down tree channel it cannot use a down cross channel or an up channel. Observe that the first channel used in any route goes from a processor to a switch, and is thus necessarily an up channel. Similarly, the last channel used in any route goes from a switch to a processor, and is thus necessarily a down tree channel.

To ensure that messages get routed properly to their destinations, the algorithm employs the concept of *extended ancestors* defined below.

Definition 1 *Let u and v be two nodes in the network. Node u is said to be an* ancestor *of node v if there exists a path from u to v consisting of only down tree channels. Node u is said to be an* extended ancestor *of node v if there exists a path from u to v consisting of zero or more down cross channels followed by zero or more down tree channels.*

Observe that if u is an ancestor of v then it is also an extended ancestor of v, but the converse is not necessarily true. Routers use this extended ancestral information to compute the set of allowable outgoing channels as follows:

1. If the incoming header enters the router on an up channel, any outgoing up channel may be used.

2. If the incoming header enters the router on an up channel or a down cross channel, any outgoing down cross channel may be used if its endpoint is an extended ancestor of the destination. In this case, only down cross channels and down tree channels may be used subsequently.

3. In all cases, a down tree channel may be used if its endpoint is an ancestor of the destination. In this case, only down tree channels may be used subsequently.

Observe that the algorithm is partially adaptive. Routing in the up subnetwork is adaptive since any up channel may be selected while the message is in the up subnetwork. When the message enters the down subnetwork, there may be multiple possible down cross or down tree channels on which the message can route. A number of possible selection functions could be used to select a channel from those provided by the routing function.

3.2 SPAM Multicast Routing

The algorithm for multicast messages is a generalization of the algorithm for unicast messages described above; it contains the unicast algorithm as a special case. A multicast message is first routed to the least common ancestor (LCA) of the set of destinations using the unicast algorithm described above. (Observe that if the message is a unicast, the LCA is the destination itself, so the multicast algorithm simply reduces to the unicast algorithm.) Once the message has arrived at the LCA, all subsequent routing is restricted to down tree channels. The head of the worm will need to split at the LCA into a multi-head worm and the heads of these multi-head worms may split repeatedly in order to reach all of the destinations.

When the head of a worm enters a router, it enqueues a request in an *output channel request queue (OCRQ)* for each of the one or more channels that it requires. A requested down tree channel can either be a channel connected to a destination processor or a channel connected to another switch. A request for a set of output channels is assumed to be an atomic operation; all of a message's requests are enqueued before any other message can enqueue requests at that router.

A message entering a router waits until all of the requests that it has made for outgoing channels have arrived at the heads of their OCRQ's and all of these channels become free. Once a message has acquired all of the output channels which it requires, the header flit of the message is replicated from the input channel to all of these output channels. When all of these output channels become free again, the next flit of the message is replicated to them. The replication step repeats until the tail flit is replicated to these output channels, at which point the channels are released and made available to the waiting messages in their respective OCRQ's. Since some output buffers reserved by a particular message may be free while others are blocked, empty bubble flits are propagated to the empty buffers allowing the heads of a message to advance independently of one another.

An example of a direct network is illustrated in Figure 1 where tree channels are indicated by solid lines, cross channels are indicated by dashed lines, and vertices are labeled with ID's. Assume that node 5 wishes to send a multicast message to nodes 8, 9, 10, and 11. The least common ancestor of these destinations is node 4. The message header is first routed from node 5 to node 4. One possible path is $5, 2, 3, 4$ since the first channel on this path is an up channel and the subsequent channels are down cross channels. The message enqueues a request at node 4 for the down tree channels to nodes 6 and 7. Once these requests reach the fronts of both OCRQ's and the channels become available, the header flit is replicated to nodes 6 and 7. The head entering node 6 enqueues a request for the down tree channels to nodes 8, 9, and 10 while the head entering node 7 enqueues a request for the down tree channel to node 11. Assume that the down tree channel to node 8 is busy while the down tree channels to nodes 9, 10, and 11 are all free. In this case, the head at node 6 does not immediately acquire all of its requested down tree channels but the head at node 7 does acquire the single requested down tree channel $(7, 11)$. Therefore, the header flit at node 6 cannot advance but the copy of this header flit at node 7 advances to node 11.

The second flit of the message is replicated from the input buffer at node 4 to the output buffers at node 4 for channels $(4, 6)$ and $(4, 7)$. This flit remains in the output buffer for channel $(4, 6)$ because the input buffer for this channel at node 6 is still holding the first flit. However, the second flit does continue to node 7 and then node 11 in the other head of the message. The third flit of the message at the input buffer at node 4 cannot be replicated because the output buffer at node 4 for channel $(4, 6)$ is occupied. Thus, bubble flits are propagated to the output buffer at node 4 for channel $(4, 7)$ until the third flit is able to advance.

Theorem 1 *SPAM is deadlock-free.*

The proof of this theorem is omitted due to space limitations. The interested reader is referred to [8].

Theorem 2 *SPAM is livelock-free.*

The proof of this theorem is omitted due to space limitations. The interested reader is referred to [8].

4 Simulation Results

In this section we describe simulation results for single multicast, multiple multicast, and combined uni-

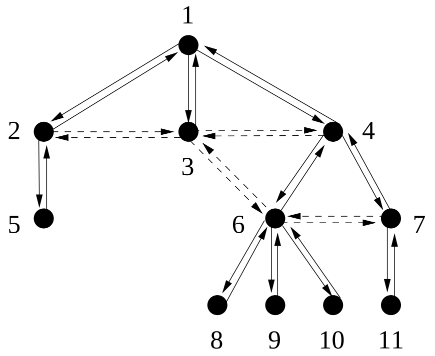

Figure 1: A symmetric network with tree edges indicated by solid lines, cross edges indicated by dashed lines, and vertices labeled with ID's.

cast and multicast traffic. The simulations were conducted using the Harvey Mudd MARS simulator, a flit-level event-driven wormhole routing simulator. Since SPAM permits partially adaptive routing, a simple selection policy was employed which prioritizes channels according to the distance from the endpoint of the channel to the LCA node.

The following system parameters were used in these experiments. Each switch was assumed to have 8 ports. In order to simulate physical proximity of connected switches, switches were randomly selected from points on an integer lattice and connected only to adjacent lattice points. Thus, at most 4 ports per switch were used for connections to other switches. In order to maximize the probability of contention between messages, each switch was connected to only one processor.

The following latency parameters were used in all experiments. The communication startup latency was 10 microseconds, router setup latency for each message header was 40 nanoseconds, and channel propagation latency was 10 nanoseconds. Each message comprised 128 flits. The measured latency for a multicast message was the total elapsed time from message startup at the source until the last flit arrived at the last destination node. Each data point in our experiments is within 1% of the mean or better, using 95% confidence intervals.

In the first set of simulations, message latency was measured for a single multicast with a varying number of destinations. These simulations were conducted for networks comprising 128 and 256 nodes. The results of these simulations are summarized in Figure 2 and clearly demonstrate that message latency is essentially independent of the number of destinations and largely independent of the size of the network. These results also confirm the advantages of hardware-supported multicast as suggested earlier by Ni and by Sivaram *et al.* [11, 14]. For example, for the latency parameters used here, SPAM incurs a latency of under 14 μsecs for a single broadcast in a 256 node network. In contrast, the theoretical lower bound for software-based multicast to d destinations is $\lceil \log_2(d+1) \rceil$ (accounting for startup latency alone), implying a lower bound of 90 μsecs in this case; a more than six-fold difference. This multiplicative factor becomes even more dramatic for larger networks.

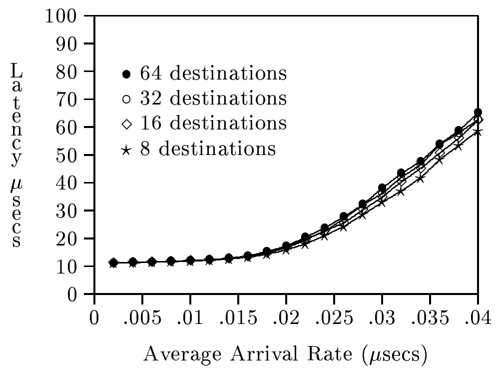

Figure 3: Latency versus average arrival rate in traffic comprising 90% unicast and 10% multicast in a 128 node network.

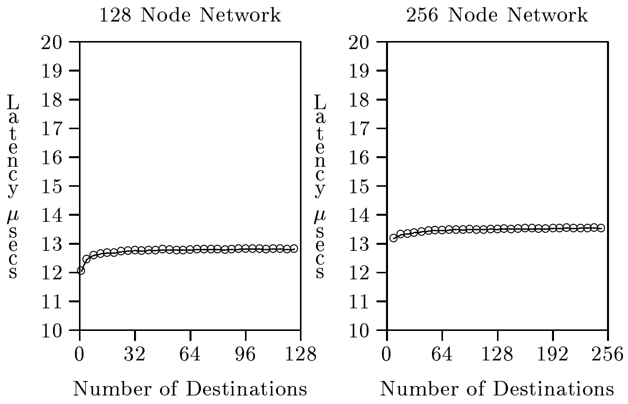

Figure 2: Latency versus number of destinations for a single multicast in 128 and 256 node networks.

In the second set of simulations, message latency was measured for mixed unicast and multicast traffic in a 128 node network in which 90% of messages were unicast and 10% of messages were multicast. Simulations were conducted for multicasts with 8, 16, 32, and 64 destinations using a negative binomial distribution with varying average arrival rates. The results of these simulations, summarized in Figure 3, demonstrate that even in relatively heavy network traffic latency remains largely independent of the number of destinations per multicast.

5 Conclusion and Directions for Future Research

In this paper we have presented a new tree-based multicasting algorithm for arbitrary interconnection topologies. To the best of our knowledge, this is the first deadlock-free tree-based wormhole routing algorithm which provides a general solution for multicasting in any direct network. Simulation results suggest that this algorithm provides a very promising means of implementing multicast with relatively simple additional routing hardware.

The theoretical technique used to prove deadlock-freedom of the SPAM algorithm is likely to have applications to the development of other tree-based multicasting algorithms for both regular and irregular topologies. In particular, for regular topologies such as meshes and n-cubes, judicious selection of spanning trees for the underlying routing algorithm may have significant effects on performance.

As noted earlier, another issue is that of utilization of input buffers that are larger than a single flit. By using larger input buffers and the input-buffer-based switching [15], for example, message latency could potentially be further reduced. Further experimental studies are required to understand the effect of increased input buffer size on message latency. The significance of the result presented here is that input buffer size can be entirely independent of message length.

Finally, SPAM uses only one worm to deliver a message to an arbitrary set of destinations. As the number of destinations increases, the probability that the worm must pass through the root of the underlying spanning tree increases, resulting in potential hot-spot effects at the root of the spanning tree; an inherent feature of the up*/down* routing algorithm. One approach to this problem is to partition the destinations into groups of contiguous nodes and send separate tree-based multicasts to each of these groups.

We are currently investigating several partitioning algorithms and their effect on message latency.

References

[1] M. Azevedo and D. Blough. Fault-tolerant clock synchronization of large multicomputers via multidimensional interactive convergence. Technical Report ECE 94-12-03, University of California, Irvine, Dec. 1994.

[2] N. Boden et al. Myrinet: A gigabet-per-second local-area network. *IEEE Micro*, 15(1):29–36, February 1995.

[3] R. Boppana, S. Chalasani, and C. Raghavendra. On multicast routing in multicomputers. In *Proc. 1994 IEEE Symp. on Parallel and Distributed Processing*, 1994.

[4] C. Chiang and L. Ni. Deadlock-free multi-head wormhole routing. In *Proc. of the First High Performance Computing Symposium, Asia*, 1995.

[5] K. Li and R. Schaefer. A hypercube shared virtual memory. In *Proc. 1989 Int. Conf. Parallel Processing*, volume 3, pages 125–132, 1989.

[6] X. Li, P. McKinley, and L. Ni. Deadlock-free multicast wormhole routing in 2-D mesh multicomputers. *IEEE Transactions on Parallel and Distributed Systems*, 5(8):793–804, Aug. 1994.

[7] R. Libeskind-Hadas, T. Hehre, A. Hutchings, Mark Reyes, and Kevin Watkins. Adaptive multicast routing in wormhole networks. In *Proc. Ninth Int. Conf. on Parallel and Distributed Systems*, Oct. 1997.

[8] R. Libeskind-Hadas, D. Mazzoni, and R. Rajagopalan. Tree-based multicasting in wormhole-routed irregular topologies. Technical Report HMC-CS-97-02, Harvey Mudd College, August 1997.

[9] M. Malumbres, J. Duato, and J. Torrellas. An efficient implementation of tree-based multicast routing for distributed shared-memory multiprocessors. In *Proc. of the the Eighth IEEE Symp. on Parallel and Distributed Processing*, pages 186–189, Oct. 1996.

[10] P. McKinley, H. Xu, A-H. Esfahanian, and L. Ni. Unicast-based multicast communication in wormhole-routed networks. *IEEE Transactions on Parallel and Distributed Systems*, 5(12):1252–1265, Dec. 1994.

[11] L. Ni. Should scalable parallel computers support efficient hardware multicast? In *Proc. of the 1995 ICPP Workshop on Challenges for Parallel Processing*, pages 2–5, August 1995.

[12] D. Panda, S. Singal, and P. Prabhakaran. Multidestination message passing mechanism conforming to base wormhole routing scheme. In *Proc. 1994 Parallel Computer Routing and Communication Workshop*, number 853 in Lecture Notes in Computer Science, pages 131–145. Springer-Verlag, 1994.

[13] M. Schroeder et al. Autonet: A high-speed, self-configuring local area network using point-to-point links. Technical Report 59, DEC SRC, April 1990.

[14] R. Sivaram, D. Panda, and C. Stunkel. Multicasting in irregular networks with cut-through switches using tree-based multidestination worms. In *Proc. of the 2nd Parallel Computing, Routing, and Communication Workshop*, June 1997.

[15] C. Stunkel, R. Sivaram, and D. Panda. Implementing multidestination worms in switch-based parallel systems: Architectural alternatives and their impact. In *Proc. of the 24th ACM Annual International Symposium on Computer Architecture*, June 1997.

[16] H. Wang and D. Blough. Tree-based fault-tolerant multicast in multicomputer networks using pipelined circuit switching. Technical Report ECE-97-05-01, Dept. of Electrical and Computer Engineering, University of California at Irvine, May 1997.

[17] H. Xu, P. McKinley, and L. Ni. Efficient implementation of barrier synchronization in wormhole-routed hypercube multicomputers. *Journal of Parallel and Distributed Computing*, 16:172–184, 1992.