Multicasting in Wormhole Routed Networks – A Critical Review

Lesley Chang
Prashant Nedungadi
Suresh Sankaralingam
Allen Ong

1.0 Abstract

Multicasting is an information dissemination problem that consists of sending a message from a single node to an arbitrary number of nodes in a network. Wormhole routing is a flow control technique that is extensively used in multi-processor interconnection networks. There have been many methods proposed for efficient multicasting using wormhole routing. This paper presents a critical review of 3 popular multicast techniques and variations to them for efficiency and deadlock avoidance. We examine Unicast-based for n-dimensional mesh topologies, Tree-based for MIN (multistage interconnection network) topologies, and Path-based for n-dimensional mesh topologies.

2.0 Problem Description

Parallel computers increase their computing speed by performing several computations concurrently, which requires coordination and synchronization between nodes through interprocessor communication. Efficient communication among nodes is crucial to the performance because regardless of how well the computations are distributed among the processors, communication overhead can bottleneck the speedup. This communication can either be unicast, multicast, or broadcast. Unicast is where a message is sent from a source node to one destination. Multicast is where the same message is delivered from a source node to an arbitrary number of destination nodes. Broadcast is a special case of multicast in which the same message is delivered to all nodes in the network.

This paper focuses on multicast communication, which has several uses in large-scale multiprocessors. These include a growing number of parallel applications, including parallel search algorithms and parallel graph algorithms. Multicast is also useful in the Single Program Multiple Data (SPMD) mode of computation, where the same program is executed on different processors with different data, specifically in several operations such as replication and barrier synchronization. Additionally, in a distributed shared-memory paradigm, multicast communication may be used to effectively support shared-memory invalidation and updating. It is clear that multicast communication has many uses, but the problem is how to implement it efficiently.

There are several metrics for evaluating the quality of a solution. One metric is its communication latency, which is the sum of three values: start-up latency, network latency, and blocking time. The start-up latency is the time required for the system to handle the message at both the source and destination nodes. The network latency is the elapsed time from when the head of a message enters the network at the source until

the tail of the message emerges from the network at the destination. The blocking time includes all possible delays encountered during the lifetime of a message. Another metric is resource requirements, such as buffer sizes, number of channels used to support a single multicast, and processor/router usage. A third metric is the computational complexity needed to use the multicast routing algorithm before the packet is injected into the network.

The constraints for this problem include the topology under consideration, the flow control method, the routing algorithm, and, in general, the system architecture. This paper focuses on systems that use wormhole routing, which reduces network latency and minimizes buffer requirements. Another important feature of wormhole routing is that the network latency is minimally affected by distance between source and destination when there is no channel contention.

The key issues for multicasting in wormhole-routed networks are latency, contentions, and deadlock. What makes the problem hard is finding the best algorithm for a given set of constraints. Multicast latency, the time interval from when the source processor begins to send the first copy of the message until the last destination processor receives the message, is clearly an indicator of multicast performance. Developing an algorithm that avoids contention would improve latency, and thus, performance. This can be tricky since developing an algorithm that efficiently implements multicasting requires a detailed understanding of potential conflicts among messages, but potential conflicts depend on the routing algorithm used. Deadlock is also an issue since the hold-and-wait property of wormhole routing makes the system more susceptible to deadlock. Next, we evaluate three different approaches to multicasting in wormhole-routed networks for parallel computers.

3.0 Critique of Techniques

3.1 Unicast-Based Multicast Communication for n-dimensional Mesh Topologies [1]

This paper suggests a technique called the U-mesh algorithm to do high performance multicasting without deadlock and contention. The idea of the paper is to find a depthwise contention-free graph between the source and the destination. Dimension-ordered routing on such a graph is guaranteed to be deadlock-free. The minimum-time implementation is of O(log n) and therefore this algorithm scales well for large networks.

The strong points of this approach are:

- The software-based multicasting approach allows us to use an existing unicast-based network and implement multicasting without any change in hardware.
- Finding a depthwise contention-free graph between source and destination while maintaining logarithmic timing is a novel approach.

Chaining Algorithm:

This algorithm is depthwise contention-free and also minimum-time implementation. The algorithm is given below:

```
Algorithm 1: The Chain Algorithm
Input: Dimension ordered address sequence
     { Dleft, Dleft+1, … , Dright},
   where Dleft is the local address.
Output: Send ⌈log2 (right – left + 1) messages
Procedure:
          p = ⌈log2(right – left + 1);
          while p > 0 do
                center = left + ⌈(right – left)/2⌉;
                D = { Dcenter+1, Dcenter+2,…, Dright};
                Send a message to node Dcenter
                with the address field D;
                right = center – 1;
                p = p – 1;
          endwhile
```

Figure 1. The chain algorithm for multicast

The algorithm requires every node between source and destination to run the algorithm, so that at each stage, the routes are calculated from the current destination. The disadvantage of this algorithm is that the chain algorithm is applicable only to those cases where the source address is less than all of the destination addresses.

U-Cube Algorithm:

The U-Cube algorithm solves the above disadvantages quoted in the chain algorithm by doing one additional step. It computes the relative destinations from the source by x-oring the destination address with its source. The new tree calculated is called the U-cube tree. One of its very interesting properties involves broadcast. The well known E-cube broadcast tree is a special case of the U-cube tree when the source and all the destinations form a subcube.

Figure 2 gives an example of this method in a 5-cube. The source node 11010 is sending to a set of six destinations {00001, 01000, 01101, 01110, 11011, 11100 }. Taking the exclusive-or of each destination address with 11010 and sorting the results produces the D0-relative dimension ordered chain Φ = { 11010, 01110, 01000, 11100, 11011, 00001, 01101 }. The corresponding U-cube tree is shown in Figure 2. It takes 3 time steps for all destination processors to receive the message.

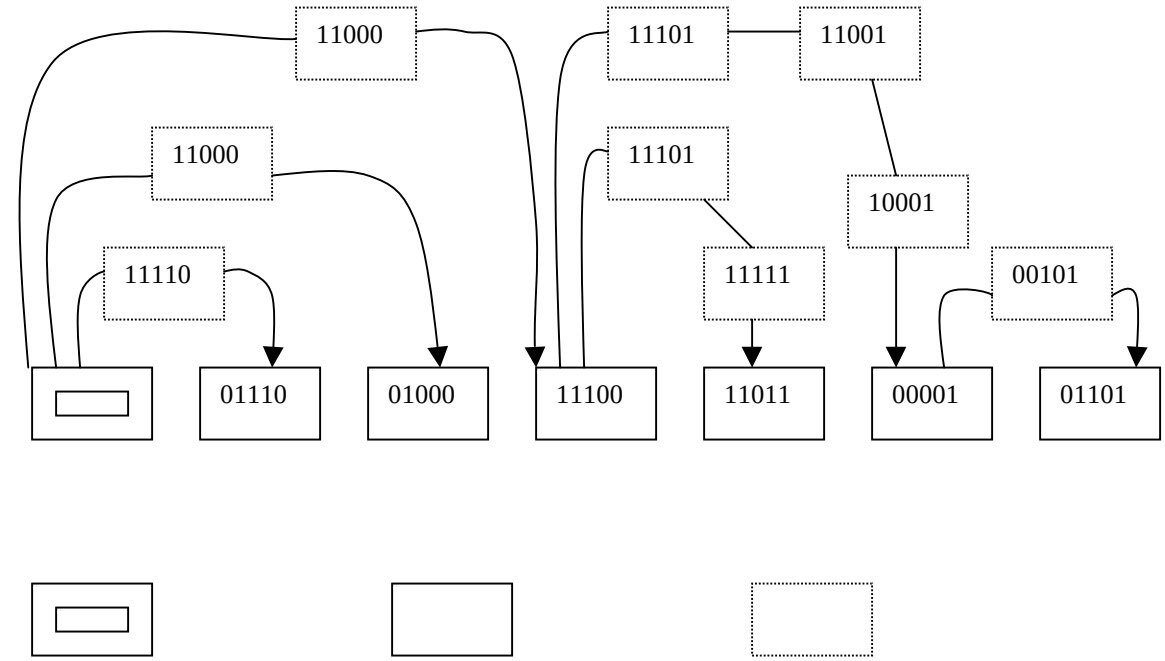


Figure 2. Multicast Chain Example in a 5-cube

Once the U-Cube tree is formed, the chaining algorithm is applied to it to get a depthwise contention-free and minimum-tree multicast network. U-Cube scales very well to higher order networks since the route computation is done by the destination nodes incrementally and not at all the nodes in the route. The source computes the route to the first destination and then each destination computes the route for the following destination. In the LEN (a distributed greedy algorithm originally designed for hardware implementation in networks using virtual cut-through switching) implementation, all the nodes along the path participate in the route calculation.

U-Mesh Algorithm:

Since n-dimensional meshes are not symmetric, the x-oring done in the U-Cube algorithm cannot be used for the mesh topology. The U-Mesh algorithm uses another variation of the chain algorithm which provides depthwise contention-free and minimum-time implementation. Here, the source and destination addresses are sorted into a dimension ordered chain, denoted by Φ, at the time when multicast is initiated by calling the U-mesh algorithm.

The source node successively divides Φ in half. If the source is in the lower half, then it sends a copy of the message to the smallest node in the upper half. That node will be responsible for delivering the message to

the other nodes in the upper half, using the same U-Mesh algorithm. If the source is in the upper half, then it sends a copy of the message to the largest node in lower half. In addition to the data, each message carries the addresses of the destinations for which the receiving node is responsible. At each step, the source deletes from Φ the receiving node and those nodes in the half not containing the source. The source continues this procedure until Φ contains only its own address.

```
Algorithm 2: U-Mesh Algorithm
Input: Φ: dimension-ordered chain
     { Dleft, Dleft+1, … , Dright},
       for source and destinations
      Dsource: the address of source node
Output:  Send ⌈log2 (right – left + 1) messages
Procedure:
          p = ⌈log2(right – left + 1);
          while p > 0 do
              center = left + ⌈(right – left)/2⌉;
              if source < center then /* send right */
                D = { Dcenter+1, Dcenter+2,…, Dright};
                right = center – 1;
              else /* send left */
                D = { Dleft,…,Dcenter-1};
                left = center;
              endif
              Send a message to node Dcenter
              with the address field D;
              p = p – 1;
          endwhile
```

Figure 3. The U-mesh algorithm

The U-Mesh Algorithm degenerates to the chain algorithm if the source happens to lie at the beginning or end of Φ. In addition, when executed at an intermediate node in the tree, the U-mesh algorithm is, again, simply the chain algorithm. The U-mesh algorithm is also depth contention-free and a minimum-time implementation.

The limitation of this model is that it is restricted to n-dimensional network topologies. The paper also makes assumptions that have the following limitations:

1. The paper assumes that wormhole routing is distance-insensitive and there is no channel contention. This assumption does not hold true since the wormhole latency increases with the distance (H * tr). Channel contention is an issue in wormhole routing with fixed number of virtual channels.

2. The paper assumes dimension-ordered routing for the routing algorithm. Dimension-ordered routing can be overloaded with many permutations of traffic and can lead to poor overall performance. So, it is not an efficient algorithm to use.

3. One-port communication architecture is adopted for the internal channel to memory communication. But, with the existing semiconductor technologies, it would be better to use multiport channels. Especially if the memory is a DRAM (which is very typical) , the latency of this model will suffer greatly.

4. With re-configurable network processors, the concept of hardware implementation to solve multicast/deadlock problems is becoming practically viable. The author's assumption about the hardware limitation does not hold true any more. The latency improvement in hardware is very significant when the network is scaled for more traffic and more number of nodes.

5. Since the separate addressing scheme, a non-efficient scheme, is a default solution to the problem, the worthiness of comparing the model with that scheme seems trivial.

3.2 Asynchronous Tree-Based Multicasting on Multistage Interconnection Networks [2]

Multistage Interconnection Networks (MIN) have been extensively studied and adopted as an interconnection fabric for multiprocessor systems. They are a class of indirect networks that provide a high degree of connectivity that lead to higher bandwidth. The performance of MINs is largely dependent on the communication latency incurred by the message when passing through the network. Delta networks are a broad sub-segment of MINs. An important property of the Delta networks is that they are self-addressing (i.e the routing path is based on the destination address). An example of a MIN is the butterfly network shown in Figure 4.

There have been many proposed solutions to address multicasting for MINs. Software-based models have relied on the underlying unicast communication without any hardware modification. The source node sends the multicast message as a series of unicast messages. The limitation of this model is that serialization of the message transfer results in increased latency. Proposed path-based approaches were not convenient for MINs because intermediate switches in MINs are not connected to any processing nodes.

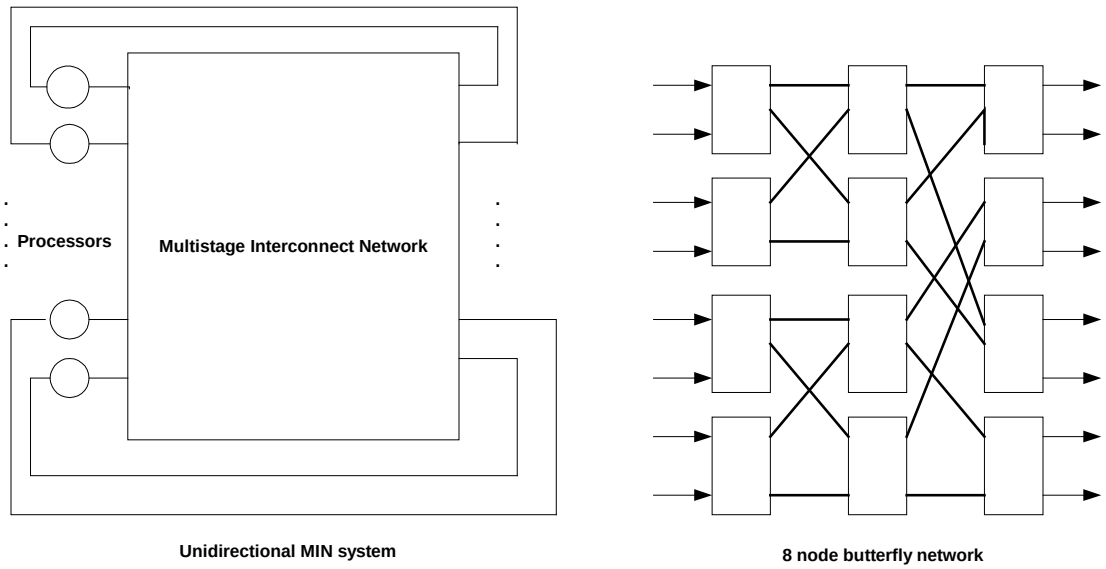


Figure 4

Synchronous Tree models proposed required the multicast headers in different branches to be forwarded synchronously with priority schemes used to prevent deadlock. The disadvantage of this scheme is that the synchronous worm needs a feedback mechanism to synchronize all headers, requiring a significant change in hardware.

This paper proposes an Asynchronous Tree Based Multicasting (ATBM) scheme for MINs that overcomes the limitations of the other proposed models.

ATBM Algorithm

The ATBM algorithm works as follows. The switches in the network are initially assigned to switching groups (explained later). When a multicast message arrives at the input port of the switch, the message header is processed. If the routing tag indicates that a tree operation needs to be performed at the current switch, the input token has to wait to gain access to the token before replicating the flit. A single token is shared among all messages in a switching group. On obtaining the token, the input port holds the token and proceeds with the tree operation. If there is another input port that is performing a tree operation, the requesting message is blocked and it waits for the other message to finish the multicast operation. While one branch is waiting for a token, the other branches can be processed asynchronously. The token is forwarded to the switches in the same group in a round-robin fashion.

Deadlock Avoidance in ATBM

Deadlock avoidance is an important issue that is addressed in ATBM. Deadlocks occur when messages acquire common buffers from a branch and request common buffers in another. A deadlock scenario can involve more than 2 messages as shown in Figure 5.

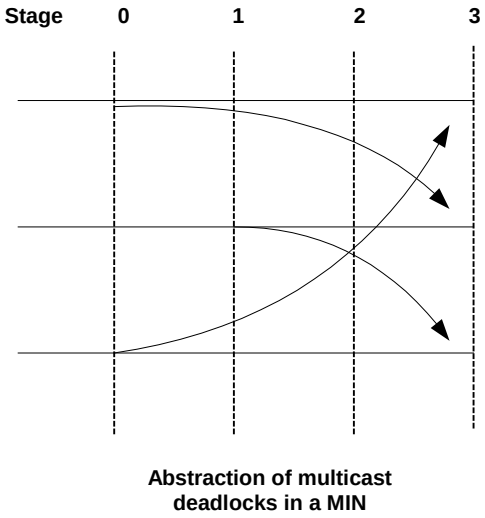


Figure 5

The following conditions are necessary for creating deadlocks in MINs:

1. Messages must belong to the same overlapped networks (switching group) to create a deadlock cycle.
2. There are at least 2 multicast messages performing tree operations in the same stage of the MIN.

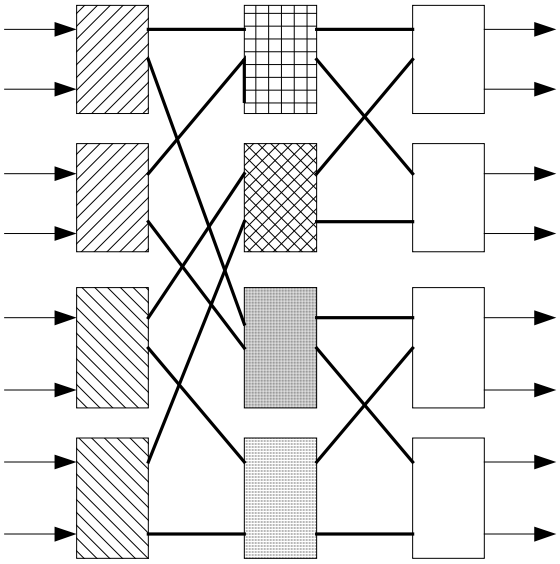
From the above condition, we can conclude that deadlocks are only possible if there are messages performing tree operations in the same switching group. The ATBM algorithm allows only one tree operation to be performed at any given instance within a switching group (by using the token passing mechanism), therefore ensuring no deadlock.

Switching groups

Switching groups are a group of switches within a stage that can potentially request the same buffer at a later stage. Switching group assignments are static and are assigned during system startup phase. In case of baseline and butterfly networks a switch on row i and stage s is assigned to the g^{th} group where

$$g = \sum_{j = 0 \text{ to } s} 2^j + i/(N/2 * 2^{s+1})$$

and N is the number of nodes in the network. Notice that the number of switches in the group decreases as the number of stages increase. Figure 6 illustrates the switching group assignment for 4-ary 3-fly baseline network.



Switching Groups in a baseline network

Figure 6

The advantage of the ATBM model is that there is very small amount of change in the hardware required (the token passing at each stage) to enable multicasting on this network. There is also a significant performance improvement over the software multicasting approaches.

The disadvantage of this model is that it is restricted to a single class of networks, the Delta networks. This approach also restricts the number of multicast operations per switching group to one (in order to avoid deadlocks). In the earlier stages of the network, the number of switches within a switching group is large, and this can potentially cause a large amount of congestion.

3.3 Path-based Multicast for n-dimensional Mesh Topologies [3]

The main motivation for path-based multicast routing algorithms is to reduce channel congestion. This is achieved by restricting the branch at each intermediate node. Unlike tree-based routing where all the required output channels have to be available in order for the packet to move forward, path-based schemes only request 1 path at a time, which leads to lower blocking probability. This becomes very attractive when the utilization and duty factor of the network links increases.

Path-based algorithms can be applied to networks that have a Hamiltonian path (visits each node in the network once and only once). It can be shown that a minimal Hamiltonian path can reach all possible nodes

in the network. However, it is NP-complete, so incrementing the number of nodes exponentially increases the time to compute the path. The algorithms require the nodes in the network to be labeled. Different label-assignment schemes are available for different topologies, such as the one given below for node (x,y) in an m x n mesh network. Two heuristic routing algorithms for multicasting were proposed based on this model.

$$\text{Node Label, } L(x,y) = \begin{cases} y * n + x & \text{if } y \text{ is even} \\ y * n + n - x - 1 & \text{if } y \text{ is odd} \end{cases}$$

The Dual-Path Multicast Routing algorithm divides the destination nodes into 2 sub-networks, D_H and D_L, where each destination node in D_H has a higher label than the source node, and D_L contains the rest of the destination nodes. Within each sub-network, the destinations are ordered based on the assigned labels, ascending in the case of D_H and descending in the case of D_L. Multicast packets are sent from the source into each of the 2 sub-networks and then propagated node-by-node as shown in Figure 7.

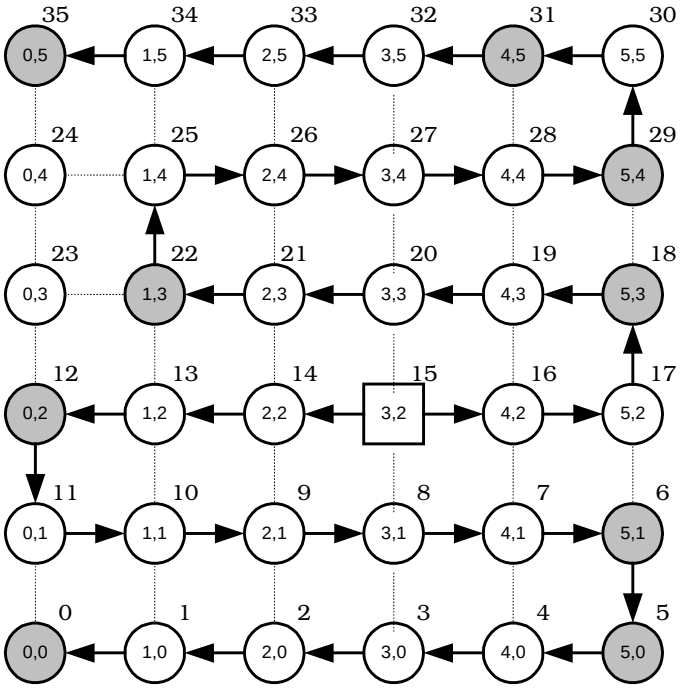


Figure 7. Example of a Dual Path Routing algorithm in a 6 x 6 mesh

The Multi-Path Multicast-Routing algorithm simply uses more sub-networks, up to the value of the outgoing degree of a node. Figure 8 illustrates the 4-path multi-path routing.

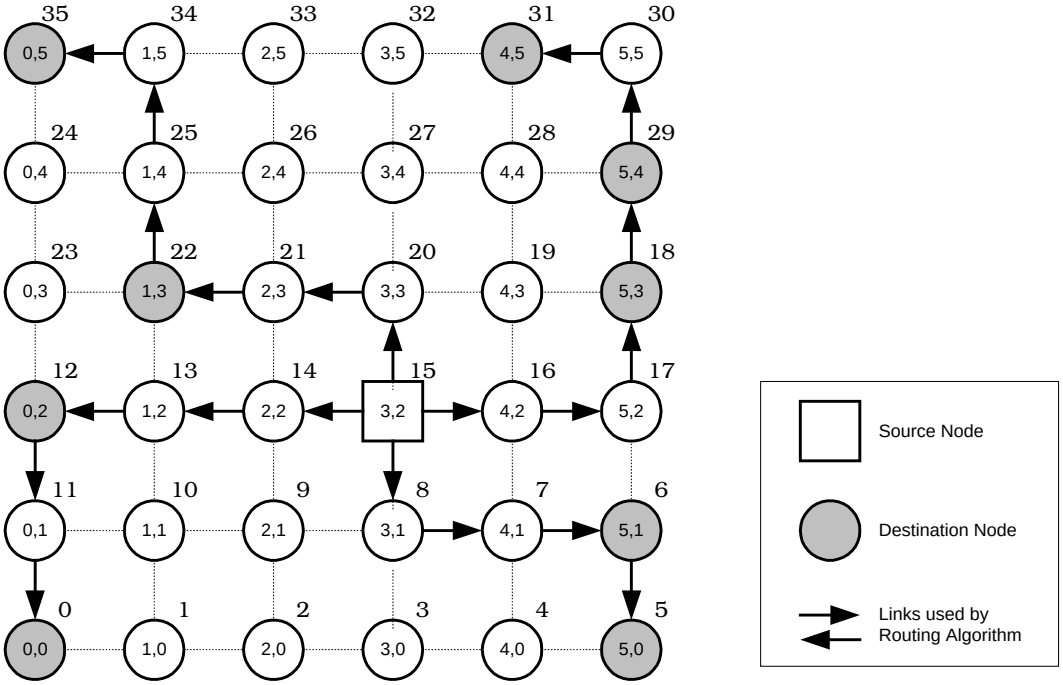


Figure 8. Example of Multipath Routing algorithm in a 6 x 6 mesh

Both of the above two schemes have been proven to be deadlock-free. Its other benefit is reducing the amount of congestion in the network with the reduced traffic. However, these routing schemes assume that the number of hops does not affect the total latency, since wormhole routing is used. This is not generally true, as stated previously. Hence, the scheme becomes non-scalable since latency is dependant on the dimensions and the number of nodes in the network. Typically, the latency across a path-based algorithm is much larger than the tree-based algorithm. The other drawback with the scheme is the strong dependency of the overall performance on the selection of an optimum Hamiltonian path. The path selection can be a complicated and time consuming for a large number of nodes.

A couple of enhancements could be suggested to improve the performance of path-based outing. One option is to add more virtual channels per physical interface to allow for more sub-networks to get to the destination. By having separate and smaller virtual channel buffers, we could minimize the intermediate node buffering delay, hence reducing the dependency of latency on the hop count. The other enhancement would be to use adaptive routing rather than deterministic to balance out the load across the unused links. This can be built upon the theoretical background for the design of a deadlock-free adaptive routing function for multicast algorithms proposed by Duato.

4.0 Recommendations

Efficient multicasting for different topologies is a hard problem and there is no single solution. The solution not only depends on the topology of the network, but also the parameters that you want to optimize (latency, contentions, and deployment cost). All of the authors have addressed the problem of making legacy systems multicast-compatible. Software-based implementations can be used in existing unicast systems, but costlier hardware-based solutions provide better performance in terms of latency. Therefore, if cost is the most important criterion, then software-based solutions are optimal.

Path-based algorithms work best on mesh topologies, if throughput is the most important criterion. In this case, the trade-off is between latency and throughput. The paths are longer, so latency is increased, but the congestion is reduced, so throughput is increased. Path-based algorithms clearly do not work for MINs since the source to destination paths are unique.

Tree-based algorithms are more suited for MINs because these algorithms more efficiently utilize the network topology. The paths from the source to multiple destinations naturally form a tree. The biggest issue in MINs using tree-based algorithms is deadlock and as long as deadlock can be avoided efficiently, this can be an optimal solution.

5.0 References

[1] P.K. McKinley, H. Xu, A.H. Esfahanian, and L.M. Ni, "Unicast-Based Multicast Communication in Wormhole-Routed Networks," in *Proceedings of the 1992 International Conference on Parallel Processing*, January 1992.

[2] V. Varavithya and P. Mohapatra, "Tree-Based Multicasting on Wormhole Routed Multistage Interconnection Networks," Technical Report TR-ACAR-96-01, Department of Electrical and Computer Engineering, Iowa State University, 1997.

[3] X. Lin and L.M. Ni, "Deadlock-Free Multicast Wormhole Routing in Multicomputer Networks," Department of Computer Science, Michigan State University, October 1990.

[4] E. Fleury and P. Fraigniaud, "Strategies for path-based multicasting in wormhole-routed meshes," Laboratoire de l'Informatique du Parallelisme, Ecole Normale Superieure de Lyon, August 1997.