Asynchronous Tree-Based Multicasting in Wormhole-Switched MINs

Vara Varavithya, Member, IEEE, and Prasant Mohapatra, Senior Member, IEEE

Abstract

Multicast operation is an important operation in multicomputer communication systems and can be used to support several collective communication operations. A significant performance improvement can be achieved by supporting multicast operations at the hardware level. In this paper, we propose an asynchronous tree-based multicasting (ATBM) technique for multistage interconnection networks (MINs). The deadlock issues in tree-based multicasting in MINs are analyzed first to examine the main cause of deadlocks. An ATBM framework is developed in which deadlocks are prevented by serializing the initiations of tree operations that have a potential to create deadlocks. These tree operations are identified through a grouping algorithm. The ATBM approach is not only simple to implement but also provides good communication performance using minimal overheads in terms of additional hardware requirements and synchronization delay. Using the ATBM framework, algorithms are developed for both unidirectional and bidirectional multistage interconnection networks. The performances of the proposed algorithms are evaluated through simulation experiments. The results indicate that the proposed hardware-based ATBM scheme reduces the communication latency when compared to the software multicasting approach proposed earlier.

Key Words: Asynchronous tree-based multicasting, Multistage interconnection networks, Deadlock configurations, Multicast routing algorithm, Wormhole switching.

AUTHOR AFFILIATIONS:

Vara Varavithya is with the Department of Electrical Engineering, King Mongkut's Institute of Technology, Bangkok 10800, Thailand. E.mail: vara@hpc.ee.kmitnb.ac.th.

Prasant Mohapatra is with the Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824. Email: prasant@cse.msu.edu.

1 Introduction

Multiprocessor systems often require coordination and synchronization between processing elements through interprocessor communication which can be either one to one (unicast communication) or could involve a group of processors (collective communication). Unicast communication is concerned with sending a message from a source node to one destination. Collective communication [1] involves a group of processing nodes that intercommunicate in a specific manner. Examples of collective communication primitives are barrier synchronization, broadcast, gather, scatter, all-gather, all to all, global reduction, and scan. Because of the nature of parallel programming, which requires a group of collaborating processors to complete a single task, efficient support of collective communications is a critical issue in the design of high performance parallel systems [2]. The inclusion of collective communications in the Message Passing Interface (MPI) standard [3] further justifies the research in this area. An important communication primitive among collective operations is the multicast communication. Multicast communication is a generalization of the broadcast operation and is concerned with sending a single message from a source node to a set of destination nodes. The multicast primitive can be also used as a basis for many other collective operations such as barrier synchronization and cache invalidations in multiprocessor systems.

Multicast communication using wormhole switching has been studied extensively for direct networks [4, 5, 6, 7, 8]. However, studies on indirect networks, such as multistage interconnection networks (MINs), have been limited [9, 10]. This paper addresses the issues associated with multicasting in wormhole-switched MINs. MINs have been extensively studied and adopted as an interconnection fabric for multiprocessor systems [11]. Examples of contemporary multiprocessors that use MIN include NEC Cenju-3 [12] and IBM SP1/SP2 [13, 14].

Wormhole switching technique has been adopted in most of the current generation systems because of its low latency. However, an efficient routing algorithm needs to be developed to exploit the advantages of wormhole routing and the interconnection topology. Most of the contemporary parallel systems are designed to efficiently support unicast communication. Efficient software multicast algorithms have been proposed for bidirectional [15] and unidirectional MINs [16]. These algorithms rely only on the underlying unicast communication and do not need any hardware modification. The source node sends unicast messages to one multicast destination in the first phase. In subsequent phases, some destinations are assigned to act like source nodes in addition to the original source node to send messages to destinations that are yet to be reached according to a predefined multicast tree structure. The number of communication phases required in binomial software multicasting schemes for d destinations is $\lceil \log_2(d+1) \rceil$. Each phase incurs a startup latency which is a major proportion of the total communication latency. In [17], a multinomial tree structure for the source and the destinations for multicasting is proposed. The multinomial multicasting steps are tuned based on the communication network parameters to further optimize the communication latency. However, the software-based approaches have a higher latency as they involve several communication start-up phases and do not exploit the concept of sending the message concurrently to cover as many destinations as possible.

To further improve multicasting performance, the multicast operations need to be supported at the lower level [18, 19]. The low-level multicast supports can be implemented as additional functions in network interface units and/or dedicated hardware in the switching elements. Hardware multicasting allows sharing of network resources to cover multiple destinations which reduces both network traffic and the number of communication phases. Hardware-based multicasting algorithms can be classified as path-based or tree-based. In the path-based approach, improvement in performance is exploited from the destinations that can share a common path. Several path-based multicasting algorithms have been proposed for the direct networks [5, 6, 7, 8]. Most of these works focus on designing deadlock-free multicast algorithms for wormhole switched networks. The path-based approach is not convenient for MINs because the messages pass through intermediate switches that are not connected to any processing node. Hence, the path-based multicasting scheme in MINs does not reduce the amount of traffic injected into the network. Also, it has been shown in [20] that the path-based multicasting approach is not only inefficient but also can create deadlocks in MINs.

MINs inherit the tree topology which can be effectively exploited to support multicast communication. Unfortunately, the tree branching operations create additional resource dependencies that make the wormhole networks more susceptible to deadlock configurations. The tree-based multicasting scheme proposed by Chiang and Ni [20] requires the multicast headers in different branches to be forwarded synchronously from one stage to the next (called synchronous worm). Synchronous movement is required to prevent deadlocks. To synchronize all multicast branches, a feedback mechanism that traverses the whole multicast tree is required. The synchronous multicasting method not only requires a considerable amount of hardware modifications but also suffers from the synchronization overhead in terms of additional latency. Another approach of tree-based multicasting is the asynchronous replication scheme in which the asynchronous worm allows multiple headers to be forwarded independent of each other. If a branch of multicast message is blocked, the other branches can be forwarded asynchronously by introducing bubbles in their paths. The asynchronous worm is preferred over the synchronous worm because of its ease of implementation. However, it is more prone to deadlocks. Deadlock prevention in asynchronous multicasting approaches is very difficult while using wormhole routing without large buffers at each of the switching elements [20]. An approach to implement the asynchronous multicasting through the use of large buffers at each switch to prevent deadlocks is reported in [21, 22].

To minimize routing operations at the intermediate switches, the multiport encoding scheme scheme [23] uses the same routing tag for different multicast branches at the same stage. This method allows limited tree operations to a set of destinations. Thus, completion of multicast operation may require several communication phases depending on the location and spread of the destinations. To prevent deadlocks, the multiport encoding scheme requires buffers at the switches corresponding to the maximum packet size to break the branch dependencies.

In this paper, we first investigate the deadlock problems associated with the tree-based multicasting in MINs. The deadlock configurations in the tree-based multicast in MINs are analyzed. A switch grouping technique is developed to analyze the behavior of the deadlocks. Based on the study, an asynchronous tree-based multicasting (ATBM) scheme is proposed for multicasting in MINs using wormhole switching technique. To prevent deadlocks, the switches are grouped using a grouping algorithm, and multiple tree operations are serialized within the groups. The serialized deadlock prevention is simpler than the synchronous replication approach [20]. The hardware modifications required for the proposed ATBM scheme is associated with the coordination of the switches in the same group. We have discussed the implementation of the ATBM algorithm for both unidirectional and bidirectional MINs. The proposed scheme is different from the previously proposed asynchronous schemes [23] in the sense that the multicast communications can be completed in a single start-up phase and only small buffers are required at each of the switches. The performance evaluation is carried out through simulation experiments. The results show that our approach performs significantly better than the software multicasting schemes while incurring low hardware overheads.

This paper is organized as follows. Section 2 presents the system model for the wormhole switched MINs. The deadlock problems in MINs and the switch grouping algorithm are explained in Section 3. The proposed ATBM algorithms are presented in Section 4. The simulation results are given in Section 5. Finally, the concluding remarks are reported in Section 6.

2 System Model

Figure 1 shows multiprocessor systems interconnected through MINs. The communication links can be either unidirectional or bidirectional. The bidirectional communication links can be viewed as two unidirectional communication links connected in opposite directions. A unidirectional MIN (UNI-MIN) system is shown in Figure 1 (a). The processing nodes are connected to the input ports of the MIN (stage zero). A message sent by a node is forwarded through the switches and reaches the receiving node via wrapped around communication links. A MIN using bidirectional communication links, called bidirectional MIN (BI-MIN), is shown in Figure 1 (b). The processor's communication channels are connected to the BI-MIN (stage zero) using bidirectional communication links. The right hand side ports are used for scalability purpose. The BI-MIN system can be scaled up by increasing the number of stages or by connecting another set of processing nodes to the system (as shown in Figure 1 (c)).

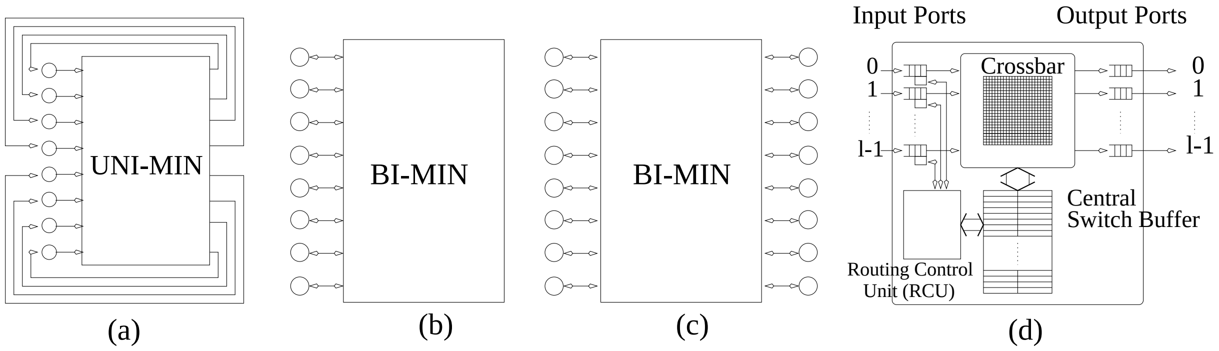


Figure 1: MIN systems (a) unidirectional MIN (b) bidirectional MIN (c) two-sided bidirectional MIN (d) switch architecture.

Switching elements are the main components of a MIN. Figure 1 (d) displays the architecture of a switch. The switch comprises of a set of input channels with the associated buffer(s), a crossbar interconnection, a central buffer space, and a set of output channels. Most contemporary systems adopt wormhole switching to reduce the communication latency. In wormhole switching [24, 25], a packet is divided into small flow control digits, called *flits*. Only the header flit(s) contains the routing information. Upon reception of the header, the control circuit examines the routing information and performs the routing operations accordingly. If the next buffer is empty, the header is forwarded to the next stage. The data flits follow the same path. After the tail flit is forwarded, the buffer is released. The pipelined flow control dramatically reduces the communication latency and is insensitive to the distance in a contention-free network. Only buffer size of one flit is required at each input channel.

A MIN using $b \times b$ switches with n stages and r rows has $N = b^n$ connection ports on each side. A switch at row i and stage j is labeled as (i, j) where $\{i \in (0, 1, \ldots, r-1)\}$ and $\{j \in (0, 1, \ldots, n-1)\}$. The node addresses are represented as $[a_{n-1} \ldots a_1 a_0]$ where $\{a_i \in (0, 1, \ldots, b-1)\}$. The source and destination node addresses are represented by $[s_{n-1} \ldots s_1 s_0]$ and $[d_{n-1} \ldots d_1 d_0]$, respectively. Figure 2 (a) shows an 8-node unidirectional baseline MIN. A bidirectional butterfly MIN with 8 nodes using 2×2 switches is shown in Figure 2 (b). The bidirectional links are depicted as two unidirectional links in opposite direction. Figure 2 (c) shows a 16 nodes butterfly MIN constructed using 4×4 switches.

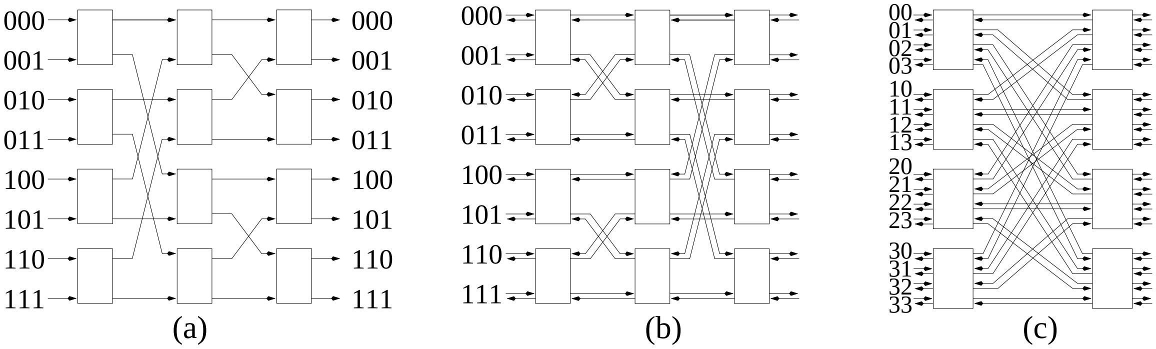


Figure 2: MIN topologies (a) 8-node baseline UNI-MIN using 2×2 switches (b) 8-node butterfly BI-MIN using 2×2 switches (c) 16-node butterfly BI-MIN using 4×4 switches.

A broad class of unidirectional MINs that belongs to the family of *Delta* networks has a self routing property by which the routing path is decided on the basis of the destination address tag [11]. Baseline and butterfly topologies belong to this class. Figure 3 (a) shows examples of destination-tag routing from [0010] ($S1$) to [1010] ($D1$), and [1100] ($S2$) to [1000] ($D2$) for the unidirectional baseline network. The next output port is decided on the basis of the destination bit corresponding to the current stage.

The routing algorithm for the unicast messages in butterfly BI-MIN uses the turnaround routing [10]. The turnaround stage T is defined as the first bit position where the s_i and d_i are different from the left hand side. The routing operation is divided into two phases. In the first phase, a message is routed freely to the stage T. After reaching stage T, the message turns around and is routed to the destination using the destination address. Figure 3 (b) shows two examples of turnaround routing from [0010] ($S1$) to [1010] ($D1$), and [1100] ($S2$) to [1000] ($D2$). From $S1$ to $D1$, the least common ancestor of the source and destination addresses is equal to 3. The message is forwarded to stage 3 and then turns back to reach its destination. Similarly, the message turns around at stage 2 for the communication between $S2$ and $D2$. Since the unicast messages request for the network resources acyclically, the turnaround routing is deadlock-free.

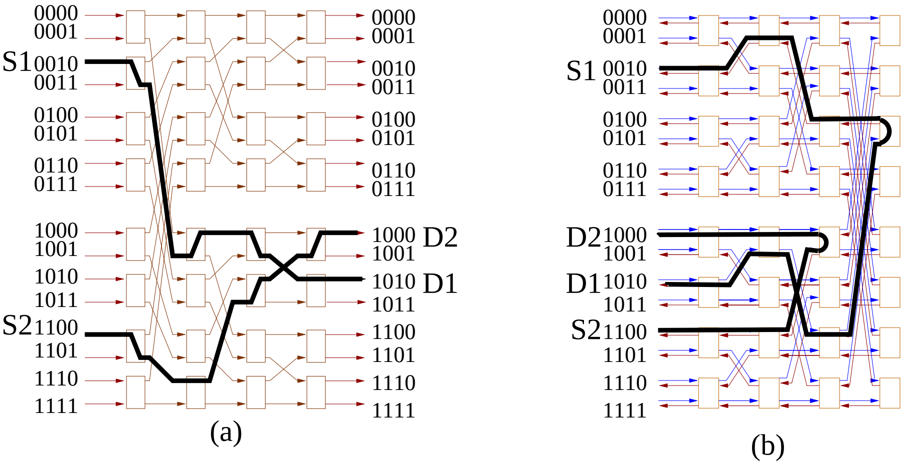


Figure 3: Routing in MINs (a) unidirectional unicast routing (b) bidirectional unicast turnaround routing.

To support tree-based multicast operations, a replication mechanism is implemented. The incoming flit is replicated and forwarded to multiple output ports in the tree operation. A flit can be copied to multiple output ports in parallel or in sequence. The switch architectures that support tree operations have been studied in [26]. Multiple headers created from the tree operation can be forwarded synchronously (synchronous worm) or asynchronously (asynchronous worm) [20]. All headers in the multicast operation are advanced concurrently at the same stage in a synchronous worm. If one header is blocked, all headers in the other branches are stalled. In contrast, the asynchronous worm allows all headers to be forwarded independently to the nodes of the multicast destination set. If one branch is blocked, the back-pressure stops the message flow at the switch that performs the replication and branch operation. Since the movement of the message stops at the tree stem, bubbles (holes) are introduced in the branches in which the headers keep moving toward their destinations. The degree of tree operations ranges from 2 to b. Figure 4 shows possible tree operations in MINs.

The tree operations are carried out as necessary at the appropriate switches. The tree-based multicasting in UNI-MINs is straightforward since it involves transmission only in one direction. The tree-based multicasting mechanism in BI-MINs can be classified based on the turnaround points [21]. Figure 5 (a) shows the tree-based multicast operation from the source (S) to a set of destinations (D) where multiple turnaround points are involved. The multiple turnaround approach uses the turnaround with forwarding to complete the multicast operation. Another method of implementation of the tree-based multicasting is shown in Figure 5 (b). In this case, a multicast message is routed to the turnaround stage and performs the tree operations in the last step of the forward routing operation. In other words, only one turnaround is allowed and tree operations are allowed only after the turnaround. This approach requires only the backward tree operations and the tree turnaround without forwarding. The amount of traffic used in the multiple turnaround scheme is lower than the single turnaround scheme. The impacts of these approaches on the deadlock prevention are discussed later in this paper.

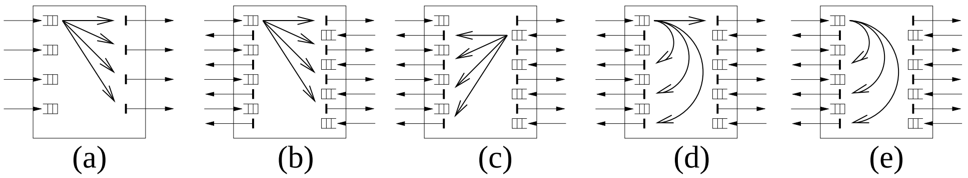


Figure 4: Multicast tree operations in MIN switches (a) The switch broadcast operation for UNI-MINs (b) The tree operation in the forward phase of BI-MIN (c) The tree operation in backward phase of BI-MIN (d) The tree turnaround with forwarding in BI-MIN (e) The tree turnaround without forwarding in BI-MIN.

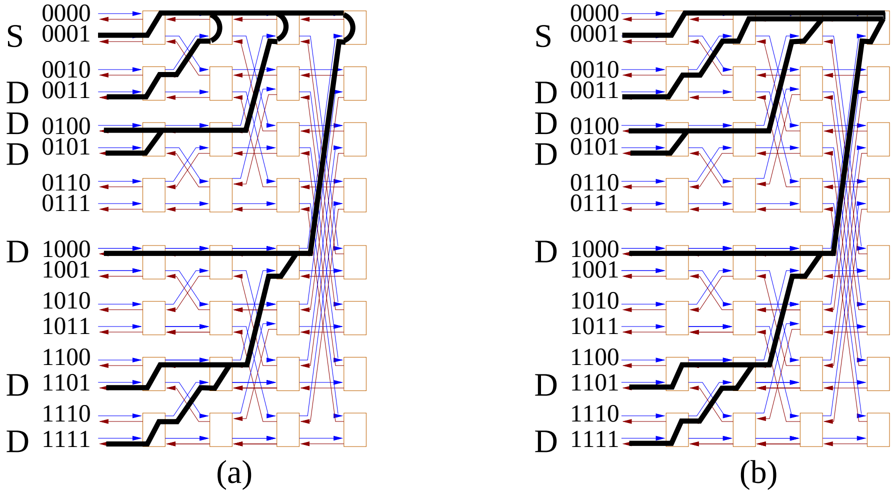


Figure 5: Multicasting in BI-MINs (a) multiple turnaround (b) single turnaround.

3 Deadlocks in Tree-Based Multicasting

The simplest form of deadlock configuration in tree-based multicasting is the single switch deadlock where the deadlock cycle involves messages at the same switch. Figure 6 (a) shows an example of the single switch deadlock configuration. Messages A and B arrive at the input ports of the switch 1. Both messages perform multicast tree operations at switch 1 at stage i. The branches request the same two buffers at switches 2 and 3 at stage $i+1$. If each branch gains an access to one buffer and requests another, a deadlock cycle is formed. If the switch 1 is in the last stage, similar deadlock cycle involving consumption channels is formed. Figure 6 (b) shows this deadlock configuration.

Another form of deadlock is created from the tree operations of the multicast messages at different switches. We term this type of configuration as multi-switch deadlock. An example of this type of deadlock is shown in Figure 6 (c). The tree operation of message A is carried out at switch 1. At a later stage, message A occupies the upper port of switch 2 and requests the lower port of the switch 4. At the same time, message B is replicated and forwarded to both the ports at the switch 3. Message B occupies the lower port of switch 4 and requests the upper port of the switch 2. Since both messages cannot proceed further due to the request-and-hold cycle of the network resources, a deadlock occurs. The single switch and multi-switch types of deadlock configurations are the basis of all deadlocks in

MINs. To illustrate the general concept behind the formation of deadlock cycles, the abstract deadlock configurations are shown in Figure 6 (d).

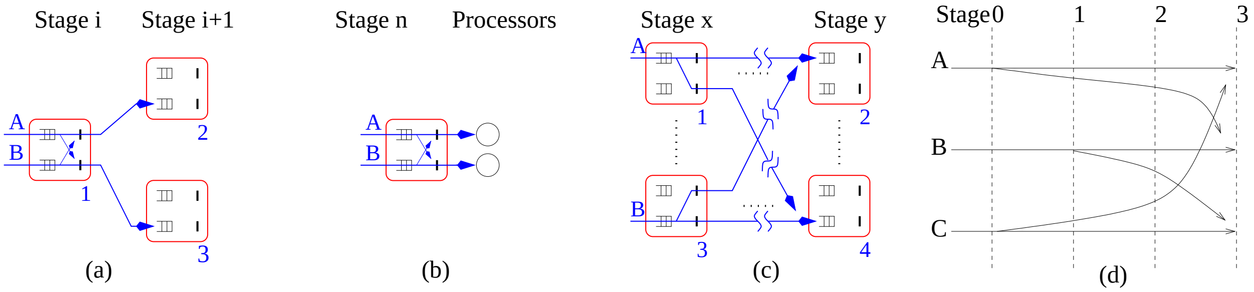


Figure 6: Deadlock configurations (a) single switch deadlock (b) single switch deadlock at the last stage (c) multi-switch deadlock (d) abstraction of multicast deadlocks.

Virtual channels [9] or dilated physical channels [11] can be used to lower the probability of the occurrence of deadlock configurations. In Figure 7 (a), the virtual channels provide alternative paths to the destinations in which the particular deadlock situation is resolved. However, if there are other multicast messages at the input buffers of switch 3, a deadlock is still possible. The number of virtual channels required is b^j at the j^{th} stage to provide a deadlock-free algorithm for UNI-MINs. This is essentially equivalent to an N-dilated MIN. Similarly, the consumption channel deadlock problem can be solved using multiple consumption channels, as shown in Figure 7 (b). To implement multiple consumption channels, the switches at the last stage must have the capability to process more than one incoming message concurrently. In direct networks, it is not uncommon to have more than one consumption channel since the router switch is closely coupled with the processor. To provide multiple consumption channels in MIN systems, special switches are needed at the last stage. If the number of consumption channels is equal to the number of input ports, the messages that arrive the last stage will eventually be consumed. The consumption channels can be either virtual consumption channels or physical consumption channels [27].

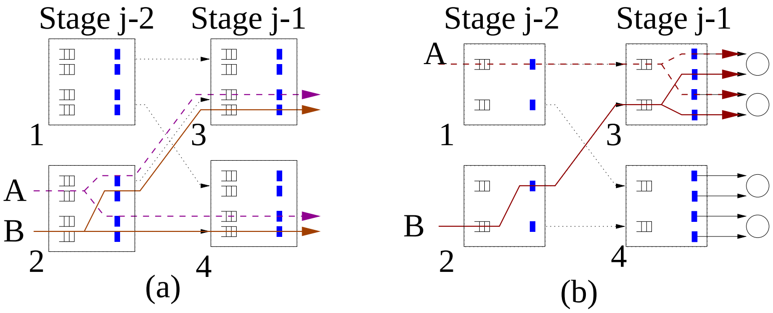


Figure 7: Deadlock avoidance (a) using virtual channels (b) using multiple consumption channels.

Deadlocks in MINs occur due to multiple tree operations of the multicast messages. However, not all combination of tree operations can lead to a deadlock cycle due to the partitionable structure of the MINs. For example, multiple tree operations at different switches (with one tree operation per switch) at the last stage are mutually exclusive. The branches created at these switches will not block each other since they all are eventually consumed by the destination nodes. Similarly, at the network level, we can partition the buffers and channels of a baseline UNI-MIN using 2 × 2 switches into two separate sets, as shown in Figure 8 (a). These two separate sets have no buffers or channels in common. Similarly, the 16 nodes with 4 × 4 switches can be partitioned in to four separate sets of buffers and channels as shown in Figure 8 (b). A message using buffers within a partition of the network will not request the buffers of any other partition in the network. Partitioning of the MIN on the basis of its topology is examined in detail in the literature [11, 28]. After partitioning the MIN, we need to group the switches of the partitions at each stage. The grouping is done in such a way that the tree operations within a group do not interact with the tree operations of any other group. Because of the special structure of baseline UNI-MINs, the grouping can be determined using following equations. The number of switches in the group N_{gj} at stage ($j \in [0, 1, \ldots, n-2]$) for the single consumption channel model is given by, $N_{gj} = b^{(n-1)-j}$ and for the b consumption channels model, $N_{gj} = b^{(n-2)-j}$. For the baseline networks, the switch (i, j) belongs to group (k, j), i.e., the switch belongs to group k at stage j. The group label (k, j) can be calculated using the equation, $k = \lfloor \frac{i}{N_{gj}} \rfloor$.

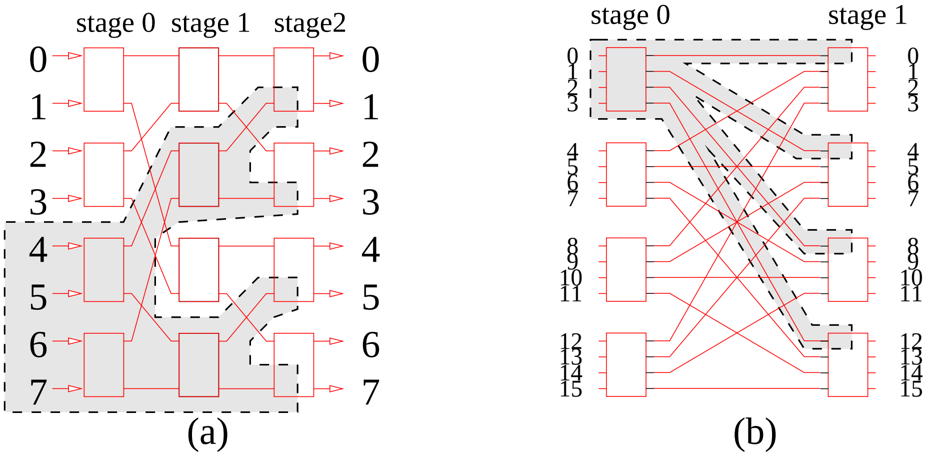


Figure 8: Partitioning of the baseline networks (a) 8 nodes using 2 × 2 switch (b) 16 nodes using 4 × 4 switch.

For other switch-based networks, the algorithm shown in Figure 9 can be used to partition switches into groups. The switches that have possible permutations to access common buffers/switches at a later stage are grouped together. The algorithm can be applied directly to unidirectional MINs. For BI-MINs, a simple network transformation is needed. The switches (i, j) in a BI-MIN are redefined as (i, j, k) where k is the new stage label. The value k is equal to j and $2n - (j + 1)$ for forward communication links and backward communication links, respectively. The BI-MIN network is partitioned into two sets based on the direction of the communication links–forward and backward networks [21]. With the reconfiguration, the BI-MIN can be considered as two concatenated unidirectional MINs as shown in Figure 10. The unidirectional MIN that sends the messages away from the source node is denoted as the forward network and the unidirectional MIN that sends the messages back to the destination nodes is denoted as the backward network. The turnaround routing operation forms the connections

```
**Switch Grouping Algorithm**
/* $b$ ports switch, $n$ stages, and $r$ rows MINs */
/* Replace $n$ by $2n$ for BI-MIN */

group_tag = $\{\emptyset\}$

begin
    for $i = 0$ to $r - 1$
        group_tag$(i, n-1) = \{i\}$; /* Replace $n-1$ by $n-2$ for $b$ consumption channels models */
    for $j = n-1$ to 0 /* Start $j$ iteration from $n-2$ for $b$ consumption channels models */
        for $i = 0$ to $r - 1$
            for $k = 0$ to $b - 1$
                $l$ = row number of the next stage switch from
                    the permutation of port $k$;
                group_tag$(i, j)$ = group_tag$(i, j) \cup$ group_tag$(l, j+1)$;
    Group the switches with the same group tag;
end
```

Figure 9: Switch grouping algorithm.

between the forward and backward networks. Note that, in some cases, the messages can turn around at the early stages.

The grouping algorithm, shown in Figure 9, assigns a group tag to each and every switch in the network. The group tag is a set of row numbers. First, the group tag of switches that are connected to the input ports of the processing nodes are assigned to their own row numbers. These group tags are propagated back along the communication links to the previous stage. The group tags of the previous stages are marked as the union of the group tags propagated back from the output ports. This process is recursively carried out until the 0^{th} stage of the forward network is reached. The switches that have the same group tag are considered to be in the same group. An example of the switch grouping is shown in Figure 10. Each switch at stage 5 of the backward network constitutes a group. The switches $(0, 1, 3)$ and $(1, 1, 3)$ have the same group tag and are considered to be in the same group. All switches in the forward network and the stage 3 of backward networks belong to the same group. This grouping algorithm can be applied to all of the MIN topologies.

If the tree operations are performed by multiple multicast messages in the same switch group, there is a possibility that a deadlock configuration could be formed. This is because, the switches in the same group can request the same set of buffers at a later stage. The tree operations from multiple mulitcast messages that initiate from different groups will never create any deadlock configuration.

The switch grouping technique for the systems with b consumption channels is sligthly different from the algorithm in Figure 9. The grouping process starts from the stage $(n-2)$. The switches at the last stage do not need to be grouped since they cannot be invloved in a deadlock configuration. The dashed line in Figure 11 shows an example of switch grouping at each stage of the baseline network. The grouping example for the single consumption channel model is shown in Figure 11 (a) for 64 nodes using 4×4 switches. For the single consumption channel model, the switches at the first stage (stage

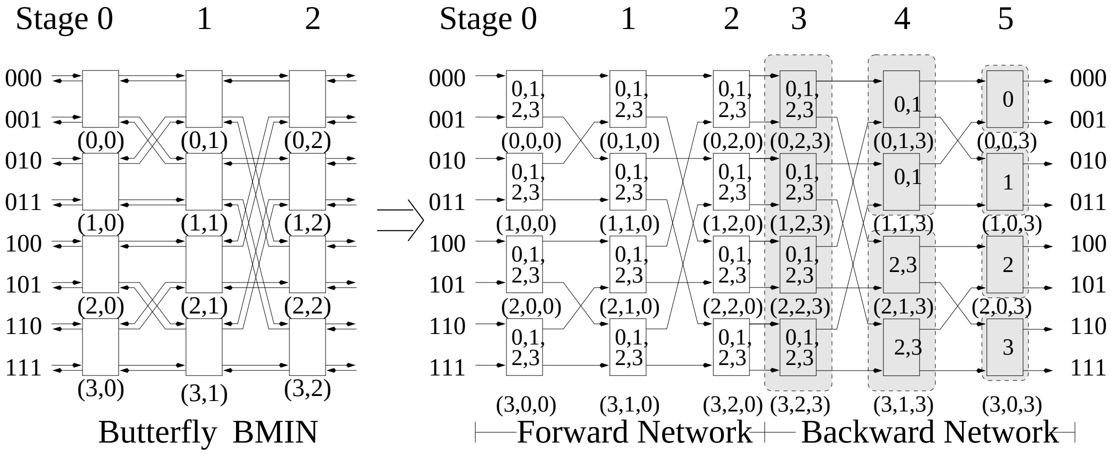


Figure 10: Partitioning of BI-MIN into forward and backward networks.

zero) belong to the same group. Notice that the number of switches in a group decreases as the number of stages increases. Figure 11 (b) shows the grouping for b consumption channels model. It is important to note that the number of switches in the group at each stage for b consumption channels model is significantly less than the single consumption therefore it allow more concurrent tree operations. All MIN systems proposed so far support the consumption of a single message in a cycle. However, we have analyzed and evaluated the cases for both single and multiple consumption channels.

4 The Proposed Algorithms

In the tree-based multicasting approach, the tree operations created by multiple multicast messages are the primary causes of the deadlock configurations. Deadlocks can be prevented either through the use of additional network resources or by controlling the routing behavior. Additional buffer space or additional physical/virtual channels can be used to prevent deadlocks. Several tree-based multicast algorithms [21, 22, 23, 29] use additional buffer space for deadlock prevention. The maximum packet size in these systems is thus limited. Packetization at the source and reassembly at the destinations are required if the actual message is larger than the limited packet size.

Control of the routing behavior is another approach for deadlock prevention. The single switch deadlock can be prevented locally at the switch using some priority schemes [20]. The multi-switch deadlock requires a complicated deadlock prevention mechanism. A synchronous worm could prevent the multi-switch deadlock configurations by restricting the multiple branches to be forwarded synchronously at the same stage. If one branch is blocked, all the other branches cannot be forwarded to the next stage. Thus, the single switch deadlock can be prevented using the priority mechanism and the multi-switch deadlock can be eliminated using the synchronous worm. However, in asynchronous routing, it is difficult to control the routing behavior to prevent deadlock configurations. It is more prone to deadlocks since the branches are allowed to be forwarded independent of the multicast destinations. We have analyzed this problem and have proposed an effective solution.

In this section, we propose a deadlock prevention technique for asynchronous tree-based multicasting (ATBM) in MINs. We have also discuss the use of this technique for UNI-MINs as well as BI-MINs.

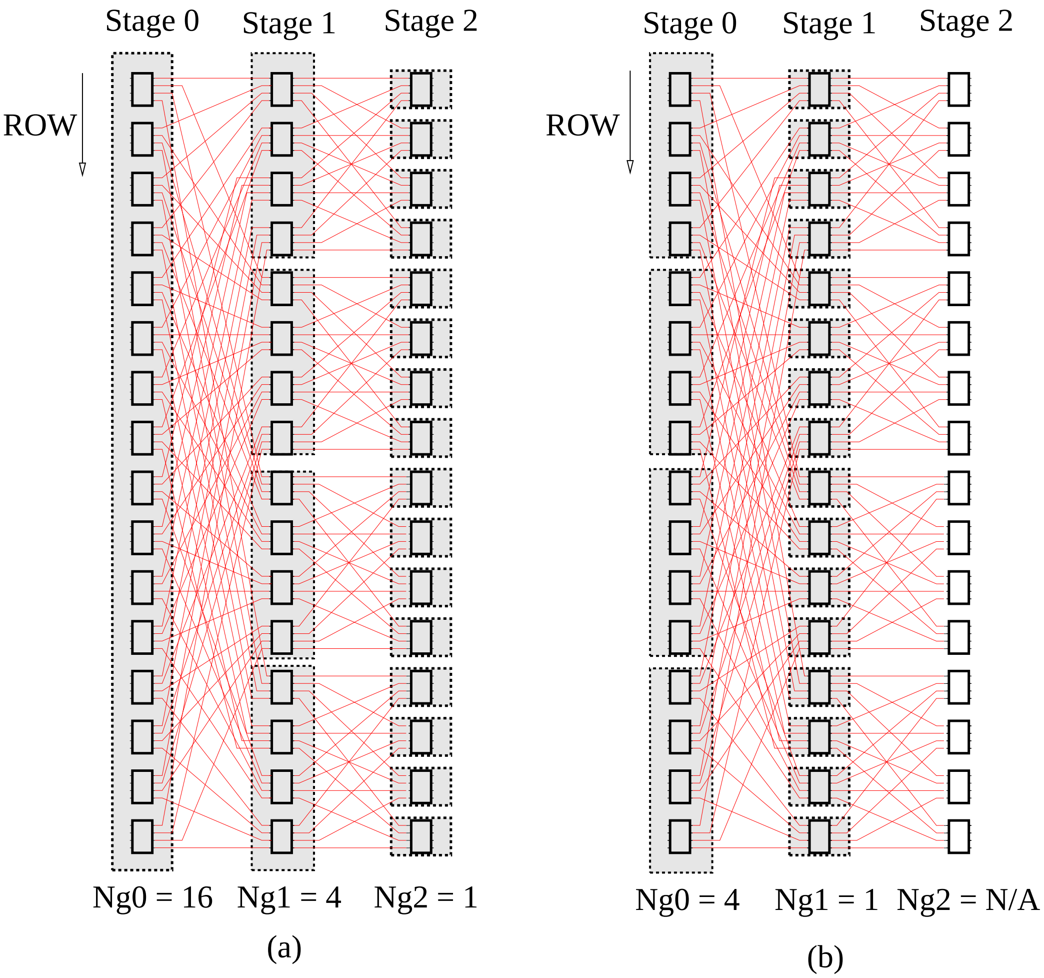


Figure 11: Switch grouping of the in a 4 × 4-switches 64 nodes baseline network (a) one consumption channel model (b) b consumption channel model.

4.1 ATBM Framework

In the previous section, we examined the main causes of deadlocks in MINs. We observed that one way of preventing deadlocks in asynchronous MINs is by avoiding the operations that have a potential to create deadlock configurations. This simple concept motivates the key idea behind the proposed ATBM scheme.

In the proposed ATBM scheme, the multicast routing information is encoded in the message header. The tree-operations are performed at the appropriate switches to distribute the multicast messages to the destinations. Multiple branches of a multicast message are forwarded asynchronously. However, the ATBM scheme imposes certain restrictions. The switches of the MIN are grouped first using the grouping algorithm presented in Section 3. As noted earlier, concurrent tree operations within a group could lead to a deadlock configuration. The proposed ATBM scheme prevents such occurrences by serializing the tree operations within a switch group. The imposition of serialization is temporary and involves only the messages that are branching at the same stage within a group. Once the multicast headers of a message reach their destinations, a subsequent tree operation (if blocked in the same group) can be initiated. The time at which a message header reaches the destination can be detected at the switch as described next. The tree operations of different multicast messages in different groups of switches are allowed to proceed without any restrictions.

The ATBM framework required only a small buffer space that can store the header of a multicast message at each input ports. There is no limit on the maximum message size that can be multicasted. Multicasting to any number of destinations can be completed in a single communication step using one communication start-up time.

Additional hardware is required in the ATBM scheme for serializing the tree operations in the switches of the same group. A control line can be implemented that interconnects the switches of a group. To preserve fairness, a hardwired token passing mechanism can be employed. The token passing mechanism is a practical approach to solve the mutual exclusion problem between switches. The deadlock recovery algorithm *DISHA* [30] has also adopted a similar token passing technique in the deadlock recovery process. The token can be passed from one switch to another through the control line in a round-robin fashion within a group. When a switch is ready to do a multicast operation, it waits until it gets the token. The switch holds the token while multicasting a message and releases it after the headers of message arrives at the destinations. Thus, the switch can relinquish the token when the number of flits that have been forwarded to the next stage is more than the sum of the buffer sizes along the path of the switches to the destinations. This value is equal to $B \times (n - j)$ where B is the buffer size at a switch, j is the current stage and n is the total number of stages. The deadlock-free property is still held because if all destinations have been reached, the multicast message will eventually be consumed by the destinations. As the number of stages in the MINs is usually less, the waiting time for the token will be within limits and multiple multicast messages can proceed if there are no blocking. This hardware modification is simpler and faster than the feedback mechanism required for synchronous multicasting [20]. The synchronous multicasting scheme needs to synchronize all multihead worms, which requires the permutation of the whole multicast tree.

In [30], the asynchronous token implementation was proposed where the token passing time is dramatically reduced. The implementation of the control line and the token passing mechanism is just an example method of implementation of the serializations. Other efficient techniques can be explored and implemented to facilitate the serialization effectively.

4.2 ATBM for UNI-MINs

Multicasting operations in a UNI-MIN can be supported ideally by an algorithm that can send a multicast message to all the destinations using a single start-up phase. This can be achieved by tree operations at the appropriate stages. We have adopted the *bit string encoding scheme* in which the routing information are represented by N bits, where N is equal to the number of nodes in the system. The bits corresponding to destinations that belong to the multicast group are set to one otherwise they are set to zero. For example, if the multicast destinations of an 8 nodes system are $\{0, 2, 5, 7\}$, the bit string for the multicast message is represented as $\{10100101\}$. The routing information in the header needs only N bits for any multicast operations in an N processors system. A minimum buffer size of

UNI-MIN Routing Algorithm(message_header)

```
1. If (unicast_message)
       Forward a flit to the output port specified in the routing_tag;
2. If (multicast_message)
       Perform routing function according to the routing table;
       If (routing function returns multiple output ports)
           Wait to obtain the token;
           Hold the token;
           Replicate and forward the header flit to the output ports supplied
           by the routing function;
           Release the token after the header flits arrive at the destination(s);
       else
           Forward the header flit to the output port returned
           by the routing function;
```

Figure 12: The ATBM routing algorithm for UNI-MIN.

N bits per input port is required to store the multicast message header in the ATBM scheme. If the routing table indicates that a tree operation need to be performed at the current switch, the multicast message has to wait to gain access to the token before replicating the flits. After obtaining the token, the input port holds the token and proceeds with the tree operation by forwarding independent headers to different output ports. If there is another multicast message requesting for a tree operation from the same switch group, the requesting message is blocked until the other message releases the token. While one branch of a multicast message is blocked either waiting for the token or by other messages, the other branches can proceed asynchronously. Bubbles are introduced in the branches that proceed asynchronously as discussed earlier by Chiang and Ni [20]. The formal description of the algorithm is given in Figure 12.

4.3 ATBM for BI-MINs

In BI-MINs, it is possible that there are more than one alternative paths from the source to the destination. The unicast message can be adaptively routed to the turnaround stage and can use the destination tag routing in the backward network to reach its destination. For multicasting, there are two choices of turnaround operations, single turnaround or multiple turnaround. Even though multiple turnaround approach uses less network resources in terms of the communication links, it is not convenient for the ATBM approach for two reasons. First, multiple tree-operations of the same message could be necessary within a switch group. It is difficult to identify such a scenario. Since all switches in the forward network will constitute of a single group, the serialization overhead is higher. Second, more information regarding multiple turnaround stages need to be contained in the header which will lead to a higher overhead. These problems do not exist in the single turnaround approach where the switches of a group are at the same stage and there is only one turnaround point. For the single turnaround, the least common ancestor of the source address and all destination addresses can

```
BI-MIN Routing Algorithm(message_header)

1. If (unicast_message)
       if (the message is in the backward network) or (current_stage = turnaround_stage)
           Forward a flit to the backward output port specified in the routing_tag;
       else
           Forward to the available output port in the forward network;
2. If (multicast_message)
       if (the message is in the backward network) or (current_stage = turnaround_stage)
           Perform routing function according to the routing table;
           If (routing function returns multiple output ports)
               Wait to obtain the token;
               Hold the token;
               Replicate and forward the header flit to the specified backward output ports;
               Release the token after the header flits arrive at the destinations;
           else
               Forward the header flit to the specified backward output;
       else
           Forward to the available output port in the forward network;
```

Figure 13: The ATBM routing algorithm for BI-MINs.

be computed in a similar manner. The multicast turnaround stage T is defined as the first position where the s_i and d_i of all multicast destinations are different from the left hand side. The multicast message is adaptively routed to the stage T and performs tree operations to forward multiple messages to all of the multicast destinations.

A routing table can be used to support the multicast tree operations in the backward routing phase. Each multicast destination is represented by a single bit string in the header. The output port(s) to which the message needs to be forwarded at the current switch is obtained by comparing the encoded destinations and the routing table. This approach enables multicasting to arbitrary number of destinations in one communication phase. The routing table can be initialized during the system start-up and is static during the system operation.

The formal ATBM algorithm for BI-MIN is shown in Figure 13. The conventional turnaround routing is performed for the unicast message. The destination tag routing is used if the message turns around at current stage or if it is already in the backward network. Otherwise, the message is forwarded to the first available port in the forward network. The multicast message performs similar turnaround routing operations. A multicast message is first checked to determine if it needs to be routed in the backward network. The backward output port(s) is supplied by the routing function (in bit string encoding scheme). If tree operations are performed (multiple output ports), the switch must wait for the token within the switch group. After the token is received, the tree operation is initiated. The token is released when all the multicast destinations have received the message header. The multicast message is adaptively routed in the forward network before the turnaround stage. Only one turn around point is allowed while using this algorithm.

5 Performance Evaluation

To investigate the effect of the proposed multicast routing algorithm, we have developed a wormhole switching network simulator. The simulation environment and the performance results are described in this section.

5.1 Simulation Model

A flit-level wormhole routing simulator was designed to implement wormhole switching networks using the CSIM event driven simulator platform [31]. We have simulated baseline UNI-MINs and butterfly BI-MINs. A variety of MIN sizes using 2x2, 4x4, and 8x8 switches were considered. The network parameters were set similar to the current technology trend. We have assumed 3.2Gbits per second communication links. The buffer size at the switches is assumed to be enough to store only the header of the incoming messages. The transmission time of one flit between a pair of switches is assumed to be 20 nanoseconds. The routing decisions are assumed to take 60 nanoseconds. The message startup time is set to 0.5 microsecond. Reception latency is ignored. Unless explicitly specified, the message size is assumed to be equal to 64 flits for both unicast and multicast messages. Each flit is assumed to be 16 bits wide. In BI-MIN routing, if multiple links are available in the forward routing phase, the lowest-number port is selected. The ATBM algorithms for both UNI-MINs and BI-MINs are compared with the previously proposed software based algorithms [16, 15]. The software based algorithm proposed for BI-MINs is unicast-based UMIN algorithm and the respective algorithm for UNI-MIN is unicast-based CMIN algorithm.

The communication latency is the time elapsed from the initiation of the message to the reception of the tail flit at the destination. The startup latency, network latency, and reception latency are components of the communication latency. The startup latency includes the message processing overhead such as the time incurred in system call and memory copying. The network latency includes the transmission delay plus the blocking delay. Finally, the destination node needs to process the incoming message and transfer it to the application. This additional time reflects on the reception latency. The multicast latency refers to the time between the message initiation and reception of the entire message by all the destinations. Both multicast and unicast communication latencies are considered as the performance metrics.

The additional overhead of the ATBM scheme is the serialization overhead of the tree operations. To simulate this behavior, additional delay is added to reflect the token passing time. The token passing mechanism is assumed to be synchronized with the switch clock. The average token passing time is equal to $20 \times N_{gj}/2$ nanoseconds, where N_{gj} is the number of switches in the group.

5.2 Performance Comparison under Contention-Free Environment

To examine the effect of the proposed scheme on multicast communication, we first analyze the ATBM algorithm under contention-free conditions. Each experiment was repeated 1000 times. The multicast source and destination nodes were selected randomly. The average multicast latency measures are plotted with respect to the number of destinations.

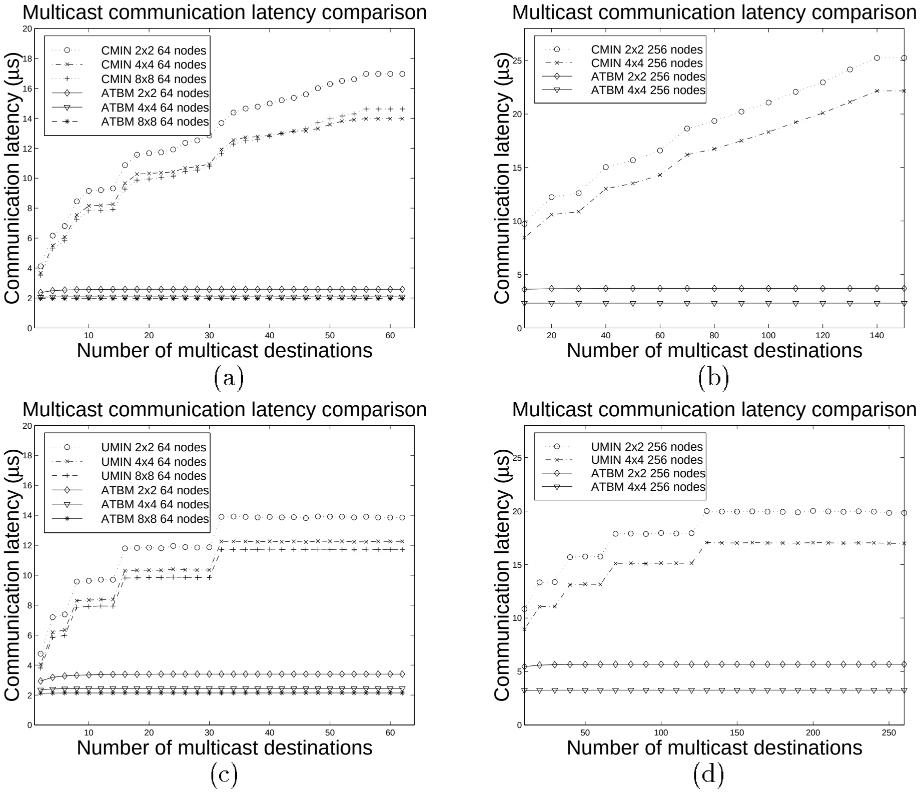


Figure 14: Performance comparison under contention-free condition (a) 64-node UNI-MINs (b) 256-node UNI-MINs (c) 64-node BI-MINs (d) 256-node BI-MINs. The multicast latency of the ATBM schemes is lower than the multicast latency of the unicast-based schemes by more than a factor of 4 when the number of destinations increases to more than one-half of the system size.

Figure 14 (a) shows the simulation results for a 64-node UNI-MIN system using 2×2, 4×4, and 8×8 switches. The performance results of the ATBM algorithms are very promising as the multicast latency using ATBM algorithm is significantly lower than the unicast-based CMIN scheme for all the cases. The multicast latency of the ATBM scheme remains almost constant when the number of destinations increases. This is because the ATBM approach requires only one communication step regardless of the number of destinations. Similar trend is observed for the 256-node system, as shown in Figure 14 (b). Figures 14 (c) and (d) show the performance results for BI-MIN systems. The ATBM approach perform quite well in BI-MIN systems under contention-free environment. T he ATBM latency is much lower than that of UMIN scheme and remains unchanged as the number of destination increases. The multicast latency of UMIN scheme increases stepwise as the number of destinations increases.

The multicast latency components of the ATBM scheme consists of startup latency, routing delay, token passing time, and transmission delay. The results from the simulation are very close to the theoretical results. For example, the 64-node system BI-MIN using 8×8 switches, the theoretical multicast latency is equal to 2.12 microseconds which is the summation of 0.5 microsecond (startup latency), 3×60 nanoseconds (routing delay), 4×20 nanoseconds (token passing delay), and $(4+64)\times20$ nanoseconds (transmission delay).

5.3 Performance of Unicast and Multicast Traffic with Contentions

The ATBM scheme serializes some of the tree operations for deadlock prevention purpose. The serialization incurs two additional overheads–serialization waiting time and token passing time. When there are several multicast operations, the initiation of tree operations might have to wait for other multicast messages to release the token. We simulate networks with contentions to study the impact of these overheads. The impact of the multicast operations on the latency of the unicast communication is also a crucial issue that needs to be considered in designing multicast algorithms. We have simulated a mixture of unicast and multicast traffic to study such an environment.

The inter-arrival time of unicast and multicast messages is assumed to be exponentially distributed. The average number of multicast destination scales up with the number of nodes in the system and we have assumed it to be $\frac{N}{2}$ with a standard deviation of $\frac{N}{4}$, where N is the number of nodes in the system. The simulated traffic is uniformly distributed among all nodes. To compare the performance in different network sizes, we have used the normalized network load to determine the inter-arrival time at each node. The network load, denoted by $Traffic$, is defined as the ratio of the average network load generated by the processing nodes in the system to the total buffer resources available in the network. The inter-arrival times for unicast (T_{arru}) and multicast messages (T_{arrm}) can be obtained using the equations,

$$T_{arru} = \frac{B_m \times N}{(B_t \times Traffic \times (1.0 - M_p))},$$

$$T_{arrm} = \frac{B_m \times M_c \times N}{(B_t \times Traffic \times M_p)},$$

where B_m is the amount of buffer-time required by a message. Buffer-time is defined as the total duration for which a message occupies buffer space in the network. M_p denotes the ratio of multicast messages generated to the unicast messages generated. M_c denotes the average number of destinations per multicast operation. B_t is the total buffer space in the network which is equal to the product of buffers at all input ports and the number of switches. Each experiment was simulated for 140,000 messages. Measurements of the first 40,000 messages were not included in the statistical results to reduce the transient effects of the network. The simulation results vary within a spectrum of 5%.

5.3.1 Simulation Results with 50% Multicast Traffic

In the first set of results, the ratio of unicast to multicast traffic is set at 50% ($M_p = 0.5$). A single consumption channel model was assumed. Figure 15 (a) shows the multicast latency versus the network load for the UNI-MIN system using 2×2 switches. The ATBM multicast latency is lower than the software multicast latency by a factor of 3 for small size systems (16-node and 64-node). For larger size system (256-node), the ATBM scheme outperforms the unicast-based CMIN scheme by a factor of 4. As the network load increases, the software-based scheme generates more traffic in the network. The message contentions lead to early saturation. In light to medium traffic conditions, the multicast latency of the ATBM scheme performs very well compared to the CMIN scheme. The effect of contention due to serialization becomes prominent with the heavy traffic. The performance results for BI-MIN are

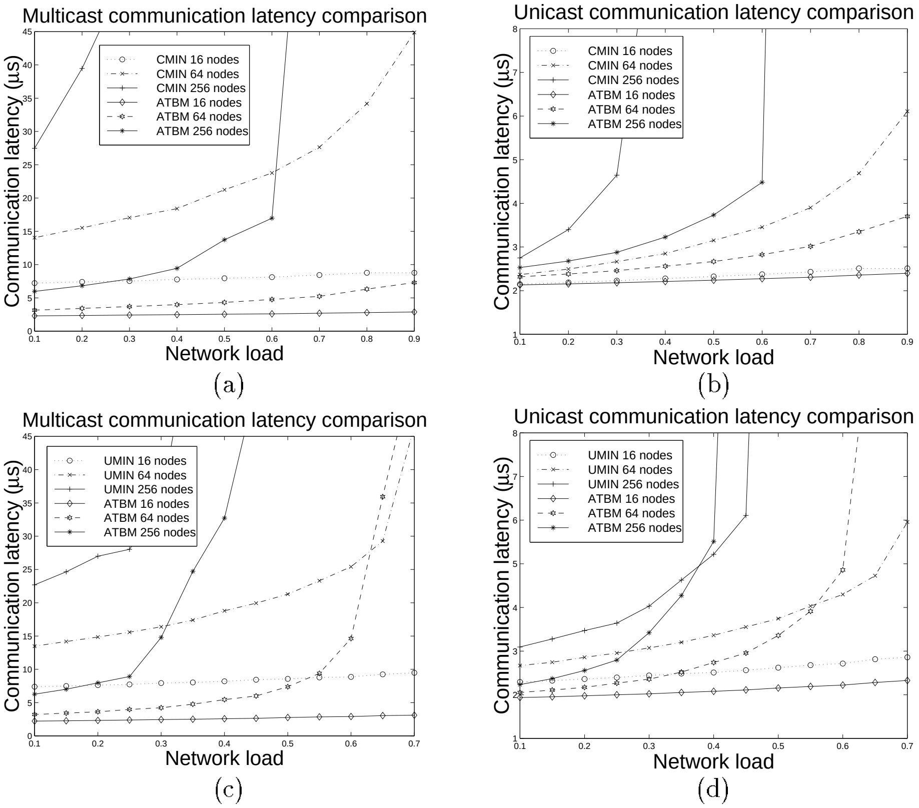
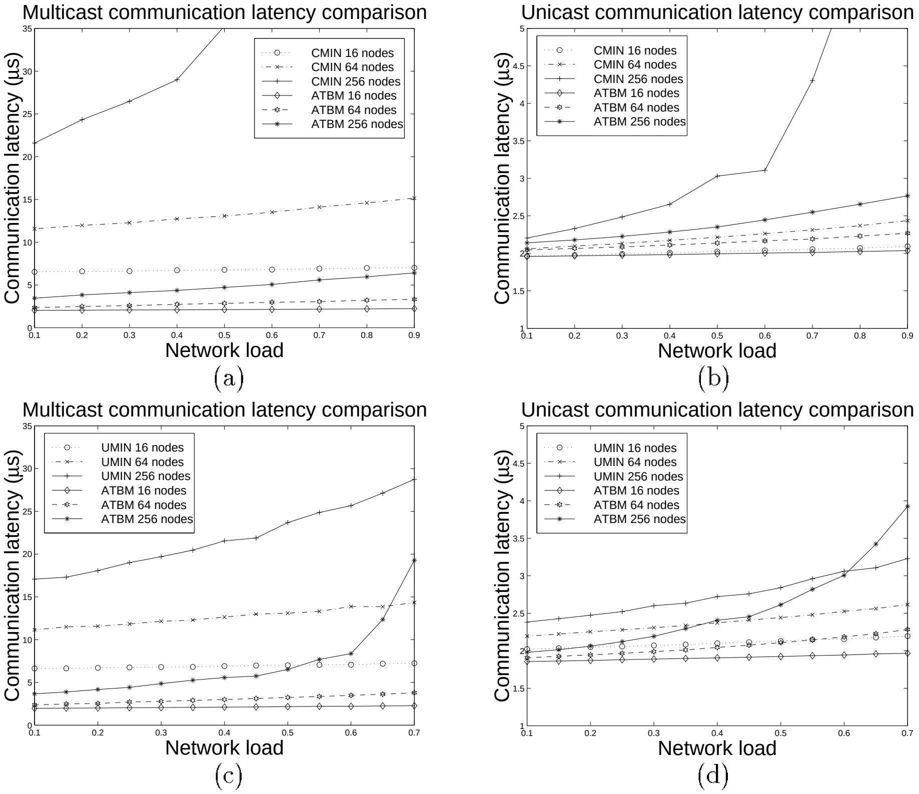


Figure 15: Performance comparison for 2×2-switch with $M_p = 0.5$ (a) multicast communication latency for UNI-MINs (b) unicast communication latency for UNI-MINs (c) multicast communication latency for BI-MINs (d) unicast communication latency for BI-MINs.

given in Figures 15 (c) and (d). Similar performance improvement for BI-MIN is achieved using ATBM approach. The ATBM performance saturate in BI-MINs earlier than that of the the UMINs in the heavy traffic condition.

With a mixture of unicast and multicast traffic, the performance of unicast communication is shown in Figure 15 (b). The multicast operations using ATBM scheme consume less network resources, therefore they impose less restrictions on the unicast communication. The unicast performance gain using ATBM scheme ranges from 25% to 40% depending on the system size with respect to the CMIN algorithm. Figure 15 (d) shows the unicast latency versus network load for BI-MIN systems. The performance improvement in BI-MIN systems is comparable to the UNI-MIN systems before the saturation point. In 64-node and 256-node systems, the unicast latency of the ATBM approach saturate slightly earlier than that of the UMIN scheme. With realistic network parameters, the overall performance of ATBM is better for both multicast and unicast communications.

As the switch size increases, the system offers more degree of connections. Contemporary MIN systems are constructed using 4×4 or 8×8 switches. With the same number of nodes, MIN systems using larger switches have lower congestion, compared to 2×2-switch systems. Figure 16 (a) shows the multicast latency versus the network load for the UNI-MIN systems using 4×4 switches. The same



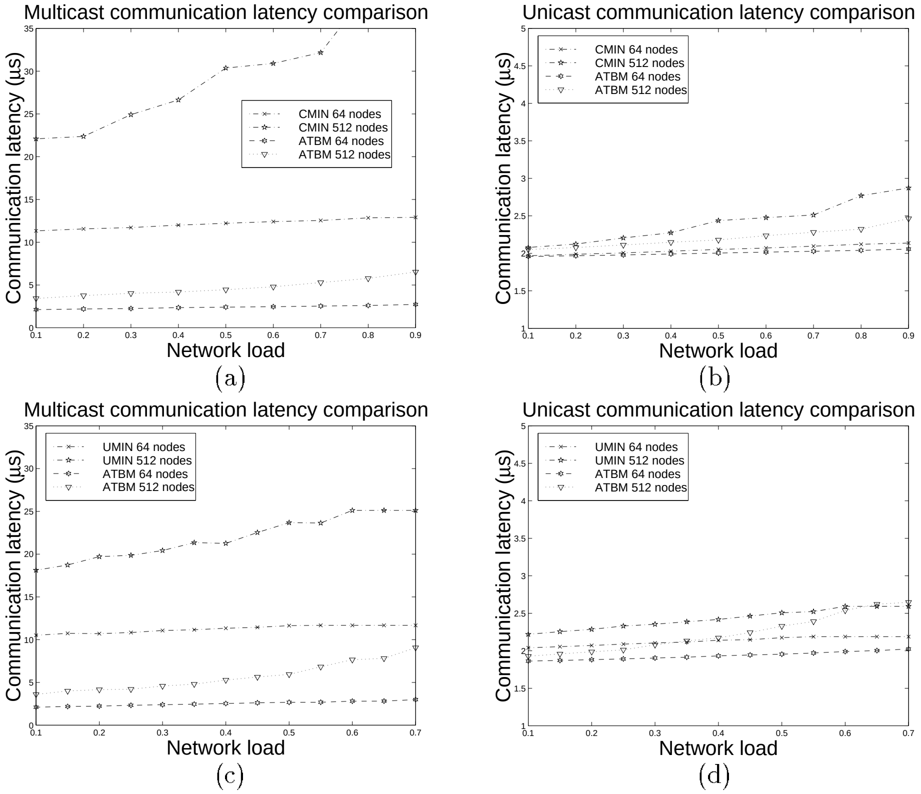
(a) (b)

(c) (d)

Figure 16: Performance comparison for 4×4-switch with $M_p = 0.5$ (a) multicast communication latency for UNI-MINs (b) unicast communication latency for UNI-MINs (c) multicast communication latency for BI-MINs (d) unicast communication latency for BI-MINs.

performance improvement is obtained in terms of the multicast latency. The contentions due to the serialization of tree operations is less as the degree of connectivity increases. For the 16-node UNI-MINs, the ATBM approach offers performance improvement of a factor of 3. The multicast latency of the ATBM algorithm is 4 times less than the software-based approach for 64-node and 256-node systems. For the UNI-MINs using 4×4 switches, the ATBM approach extends the operating range at the heavy traffic condition. In light to medium traffic region, the ATBM scheme has less impact on the unicast performance. As shown in Figure 16 (b), the unicast latency results using ATBM scheme is lower than that of the CMIN scheme. Because of the high penalty that we have imposed on the token passing time, the unicast performance of a 256-node system using the ATBM scheme degrades under heavy traffic. However, the ATBM scheme provide better performance on a wide operating range compared to the CMIN scheme. Performance comparison of ATBM and UMIN algorithms for BI-MIN systems are given in Figures 16 (c) and (d). In BI-MIN systems, the ATBM scheme shows performance improvement over unicast-based scheme in most cases.

The simulation results of the MIN systems using 8×8 switches are shown in Figure 17. These results represent systems with high degree of connections. The same level of performance improvement has been observed for both unicast and multicast communication. For the 512-node MIN systems, both unicast and multicast latency results using the ATBM scheme are significantly lower than the



(a) (b) (c) (d)

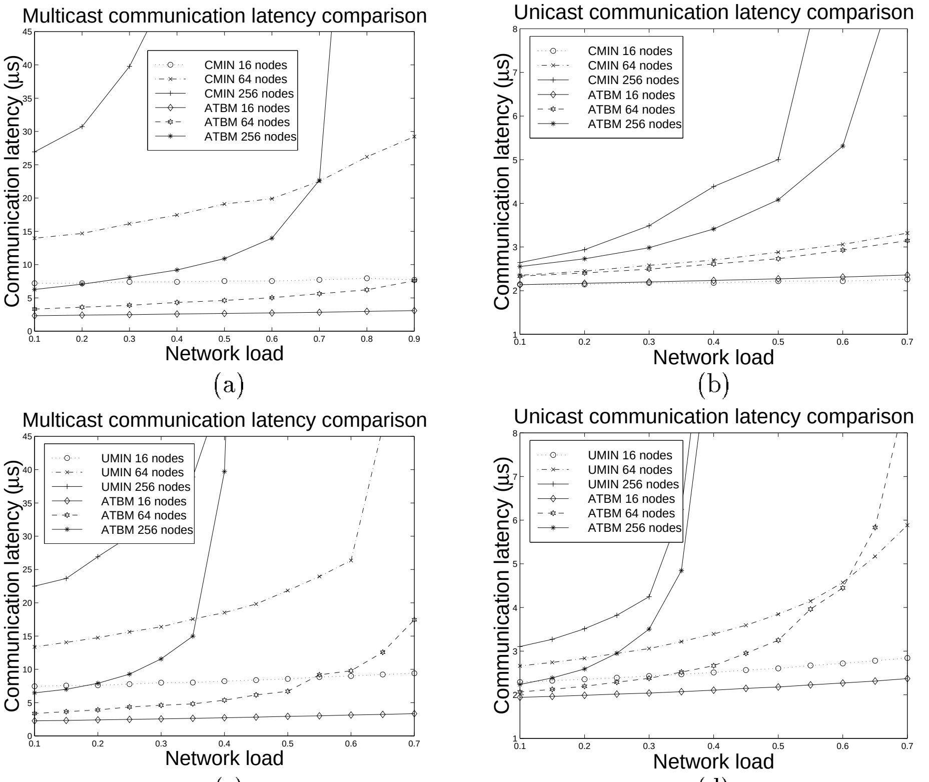
Figure 17: Performance comparison for 8×8-switch with $M_p = 0.5$ (a) multicast communication latency for UNI-MINs (b) unicast communication latency for UNI-MINs (c) multicast communication latency for BI-MINs (d) unicast communication latency for BI-MINs.

software-based scheme. This is mainly because of the fact that the traffic created while using the ATBM schemes is much less than the traffic created by the unicast-based schemes.

5.3.2 Simulation Results with 20% Multicast Traffic

In most classes of parallel applications, multicast communication constitutes only a small portion of the total network traffic. In the next set of simulation results, we assumed that the multicast traffic accounts for 20% of the total network traffic ($M_p = 0.2$). Figure 18 shows the performance comparison for the MIN systems using 2×2 switches. The comparison of Figure 18 and Figure 15 reveals the effect of the multicast traffic ratio. The latency curves for 50% multicast traffic ratio saturate earlier than the curve with 20% multicast traffic ratio. The ATBM approach performs better compared to unicast-based approach for 20%-80% ratio of multicast-unicast traffic as the operating region extends to cover higher network load.

The simulation results of MIN systems using 4×4 and 8×8 switches are shown in Figure 19 and Figure 20, respectively. With less traffic from multicast messages and the high degree of connectivity in large switches, the ATBM scheme can maintain the performance improvement ratio over a wide range of network load. The impact on the unicast communication is also reduced as shown in Figures 19 (d)



(a) (b)

(c) (d)

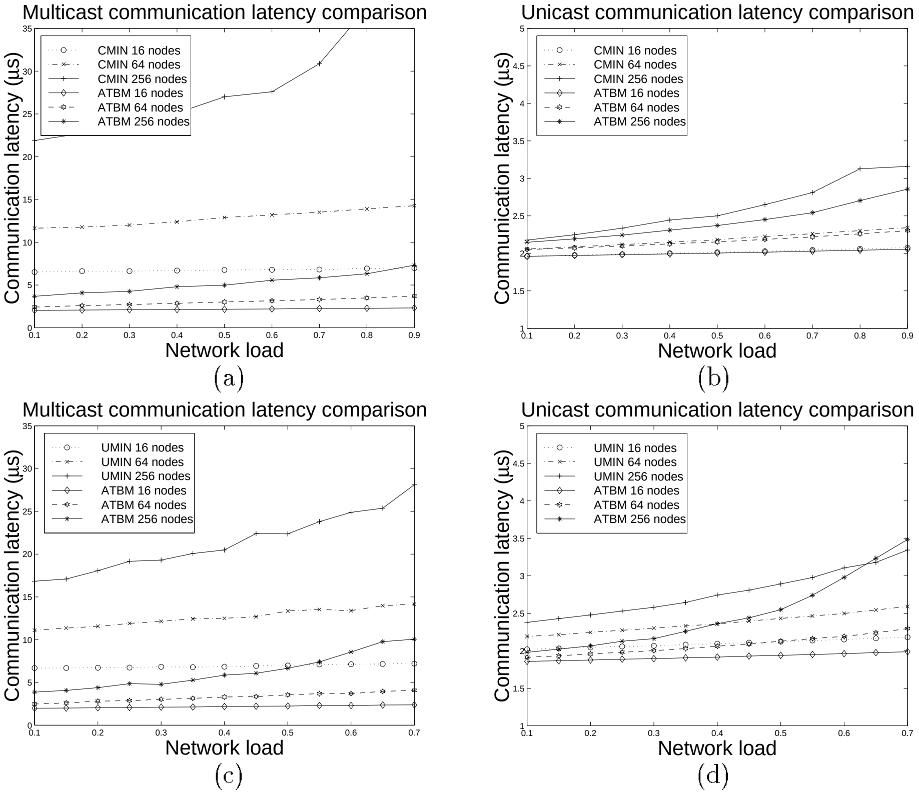
Figure 18: Performance comparison for 2×2-switch with $M_p = 0.2$ (a) multicast communication latency for UNI-MINs (b) unicast communication latency for UNI-MINs (c) multicast communication latency for BI-MINs (d) unicast communication latency for BI-MINs.

and 20 (d). The unicast latency using unicast-based UMIN scheme is higher since the network is more congested due to the multicast traffic. Even in low multicast-unicast traffic ratio, this interference of multicast to unicast is still prominent in the software-based approach.

5.3.3 Effect of Message Size

One of the major advantages of the ATBM approach is that there is no limitation on the message size. The message sizes of 128 and 256 flits were simulated to study the effect of message size using the ATBM scheme. A single consumption channel is assumed and the multicast ratio is set to 50%. Figure 21 (a) shows the latency results of 128-flit messages in a 4×4 switch-based UNI-MINs. The ATBM scheme performs quite well for the message size of 128 flits. The additional latency incurred is only from the additional transmission delay of the messages. The same observation is obtained from the systems using 8×8 switches as shown in Figure 21 (b). The results of BI-MIN systems using 4×4 switches and 8×8 switches are shown in Figures 21 (c) and (d), respectively.

The multicast latency results for 256-flit messages are shown in Figure 22. The results of UNI-MIN systems using 4×4 switches and 8×8 switches are shown in Figures 22 (a) and (b), respectively. The latency of the software-based multicast scheme increases due to the increase in transmission delay. The



(a)

(b)

(c)

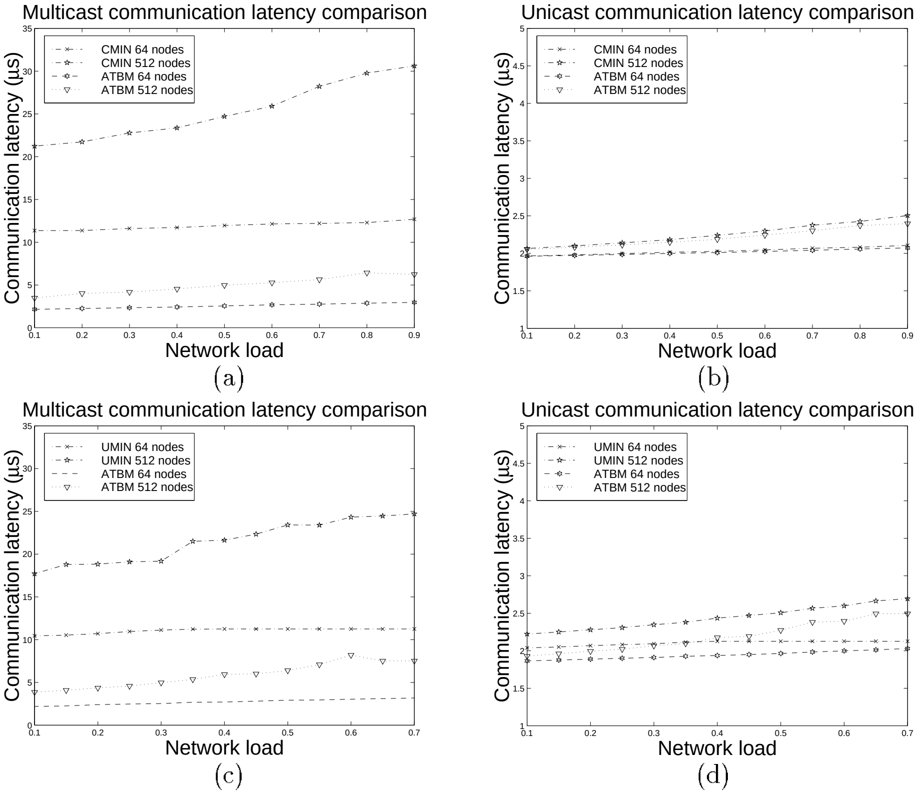
(d)

Figure 19: Performance comparison for 4×4-switch with $M_p = 0.2$ (a) multicast communication latency for UNI-MINs (b) unicast communication latency for UNI-MINs (c) multicast communication latency for BI-MINs (d) unicast communication latency for BI-MINs.

transmission delay accumulates through several phases of communications. For large size messages, the ATBM approach outperforms the software-based scheme for all system sizes. Since the ATBM approach permits the initiation of tree operations after all the headers of the current tree operation reach the destinations, non-conflicting multicast message can be routed incurring a small latency. Same level of performance improvement is observed for BI-MIN systems as shown in Figures 22 (c) and (d).

5.3.4 Evaluation of BI-MINs with b Consumption Channels

As mentioned in Section 3.2, the nodes using b consumption channels can significantly reduce the number of switches in the group which, in turn, will reduce the serialization and token passing overheads. We have simulated the MIN systems using nodes with b consumption channels with 50% multicast traffic ($M_p = 0.5$.). The multiple channels are implemented using additional consumption channels. Each consumption channel has its the communication link connected to the switch. The simulation results for MIN using 4×4 switches are shown in Figure 23. The multicast and unicast latency results for UMIN are shown in Figures 23 (a) and (b), respectively. Figures 23 (c) and (d) show the performance comparison in BI-MIN systems. Each processing node has additional hardware to concurrently receive four messages. This model eliminates the deadlock problem in the last stage switches. Both



(a)

(b)

(c)

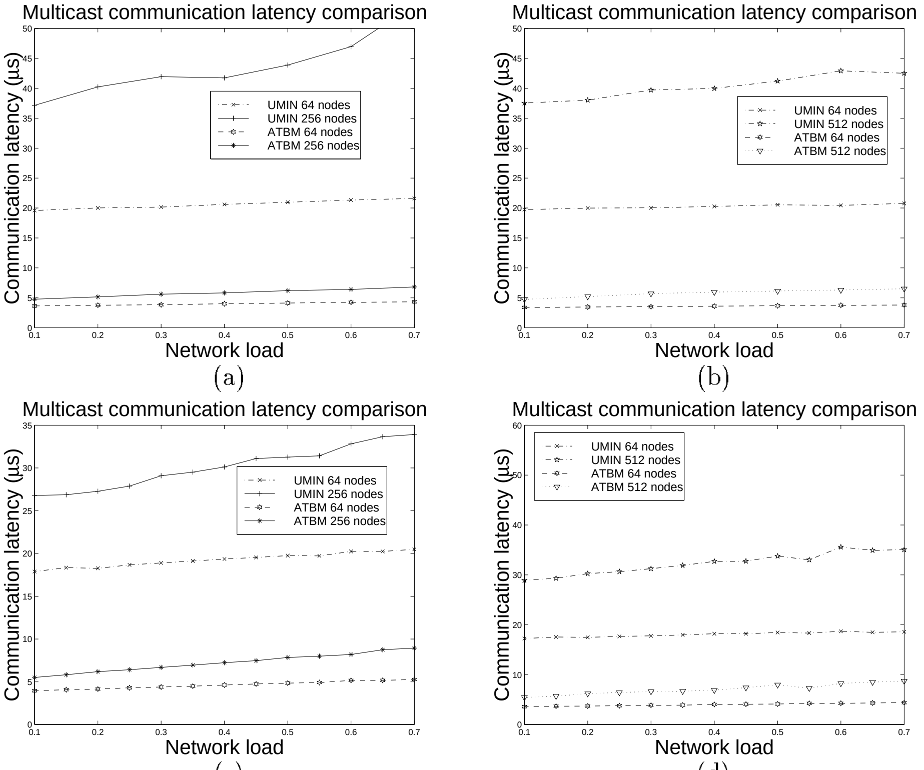
(d)

Figure 20: Performance comparison for 8×8-switch with $M_p = 0.2$ (a) multicast communication latency for UNI-MINs (b) unicast communication latency for UNI-MINs (c) multicast communication latency for BI-MINs (d) unicast communication latency for BI-MINs.

unicast-based and ATBM schemes benefit from the b consumption channels as the network throughput increases. The ATBM performance improves significantly compared to the single-port model (shown in Figure 16). With the less number of switches in the group, the serialization waiting time is less and more number of multiple tree operations are allowed to be initiated. These two factors enhance the performance of the ATBM scheme.

6 Conclusions

We have developed a framework for tree-based multicasting in MINs. The deadlock configurations that result from the tree operation are discussed. The switch grouping technique is used to analyze the potential deadlock cycles. Based on the switch groups, an asynchronous tree-based multicasting (ATBM) scheme is proposed for MINs. Deadlocks are prevented by serializing the initiations of tree operations within the same group at the same stage. We have developed complete algorithms for multicasting in unidirectional as well as bidirectional MINs. The performance of the proposed algorithm is evaluated through simulations. We have considered realistic parameters and have included the associated overheads in our experiments. The results demonstrate that the ATBM algorithm performs significantly better than the previously proposed software-based multicast routing algorithms.



(a) (b) (c) (d)

Figure 21: Multicast communication latency for 128-flit messages with $M_r = 0.5$ (a) 4×4-switch UNI-MIN systems (b) 8×8-switch UNI-MIN systems (c) 4×4-switch BI-MIN systems (d) 8×8-switch BI-MIN systems.

ACKNOWLEDGMENTS:

This research was supported in parts by the National Science Foundation through the grants MIP-9628801, CCR-9634547, and CDA-9617375. The research was done while the authors were in the Department of Electrical and Computer Engineering at Iowa State University. A preliminary version of a part of this paper appeared in the 26th International Conference on Parallel Processing (ICPP), August 1997.

References

[1] P. K. McKinley, Y. Tsai, and D. F. Robinson, "Collective communication in wormhole-routed massively parallel computers," *IEEE Computer*, vol. 28, pp. 39–50, December 1995.

[2] P. Mitra, D. Payne, L. Shuler, R. van de Geijn, and J. Watts, "Fast collective communication libraries, please," in *Proceedings of the Intel Supercomputing Users' Group Meeting*, 1995.

[3] J. J. Dongarra, S. W. Otto, M. Snir, and D. Walker, "A message passing standard for MPP and workstations," *Communications of the ACM*, vol. 39, pp. 84–90, July 1996.

[4] P. K. McKinley, H. Xu, A. Esfahanian, and L. M. Ni, "Unicast-based multicast communication in wormhole routed networks," in *Proceedings of the International Conference on Parallel Processing*, vol. 2, pp. 10–19, 1992.

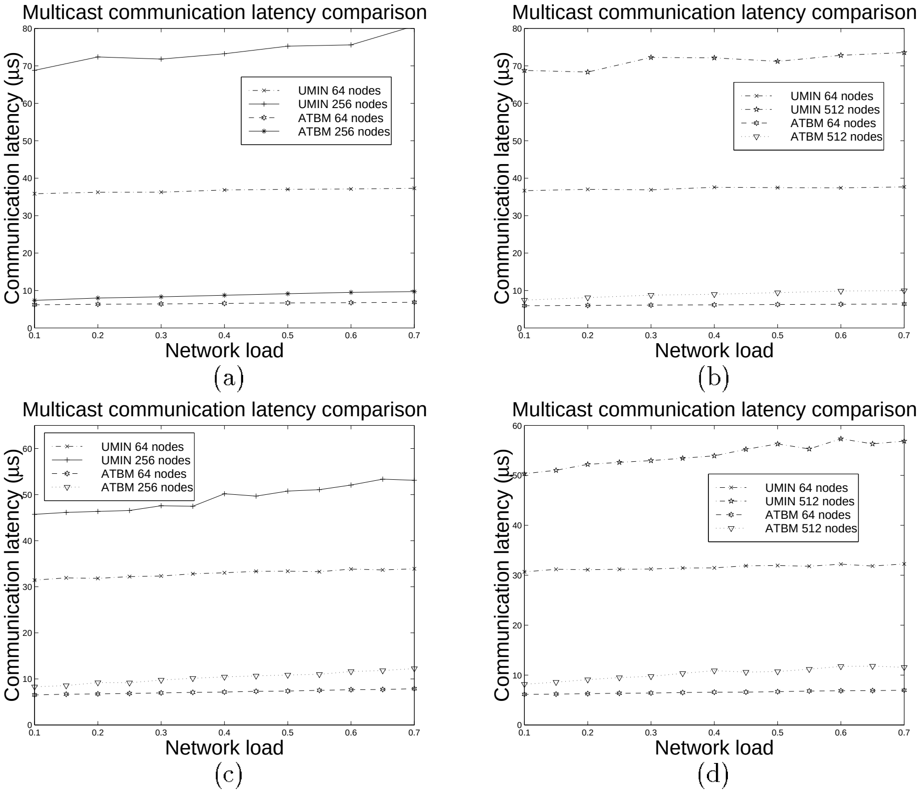


Figure 22: Multicast communication latency for 256-flit messages with $M_r = 0.5$ (a) 4×4-switch UNI-MIN systems (b) 8×8-switch UNI-MIN systems (c) 4×4-switch BI-MIN systems (d) 8×8-switch BI-MIN systems.

[5] X. Lin, P. K. McKinley, and L. M. Ni, "Deadlock-free multicast wormhole routing in 2-D mesh multicomputers," *IEEE Transactions on Parallel and Distributed Systems*, vol. 5, pp. 793–804, August 1994.

[6] R. V. Boppana, S. Chalasani, and C. S. Raghavendra, "On multicast wormhole routing in multicomputer networks," in *Proceedings of the Sixth Symposium on Parallel and Distributed Processing*, October 1994.

[7] D. K. Panda, S. Singal, and P. Prabhakaran, "Multidestination message passing mechanism conforming to base wormhole routing scheme," in *Proceedings of Parallel Computer Routing and Communication Workshop*, pp. 131–145, May 1994.

[8] P. Mohapatra and V. Varavithya, "A hardware multicast routing algorithm for two-dimensional meshes," in *Proceedings of the Eighth Symposium on Parallel and Distributed Processing*, pp. 198–205, October 1996.

[9] W. J. Dally, "Virtual channel flow control," *IEEE Transactions on Parallel and Distributed Systems*, vol. 3, pp. 194–205, March 1992.

[10] L. M. Ni, Y. Gui, and S. Moore, "Performance evaluation of switch-based wormhole networks," in *Proceedings of the International Conference on Parallel Processing*, August 1995.

[11] C. Wu and T. Feng, eds., *Tutorial: Interconnection networks for parallel and distributed processing*, (Washington, D.C.), IEEE Computer Society Press, 1984.

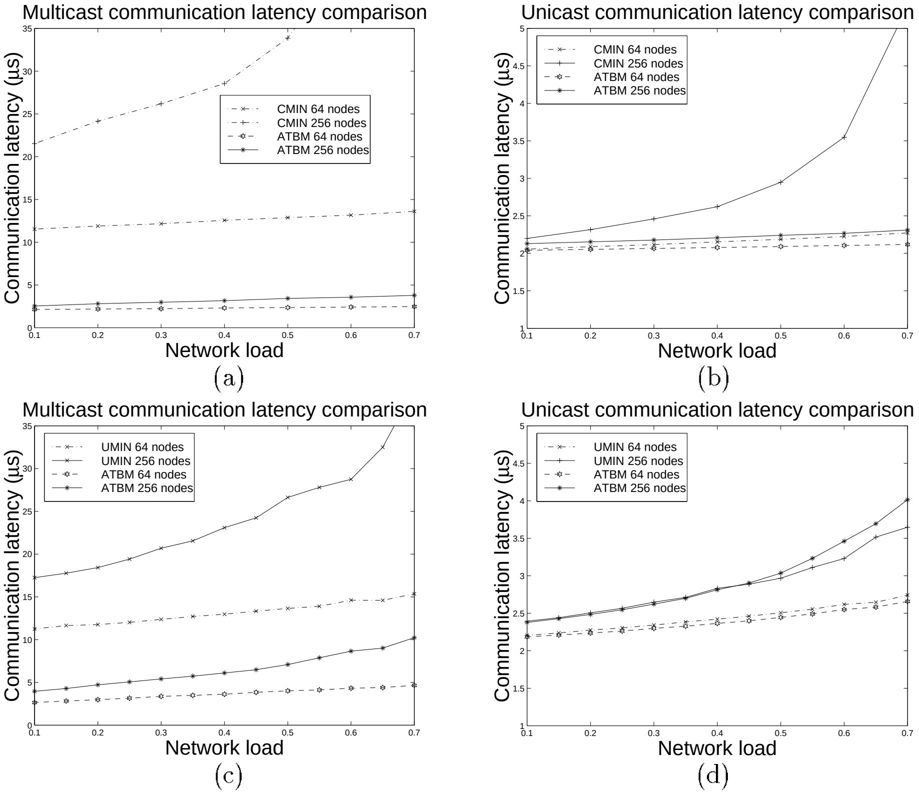


Figure 23: Performance comparison for b consumption channels model, 4×4-switch with $M_r = 0.5$ (a) multicast communication latency for UNI-MINs (b) unicast communication latency for UNI-MINs (c) multicast communication latency for BI-MINs (d) unicast communication latency for BI-MINs.

[12] N. Koike, "NEC Cenju-3," in *Proceedings of the 8th International Parallel Processing Symposium*, pp. 396–401, April 1994.

[13] C. B. Stunkel, D. G. Shea, P. H. Hochschild, and M. Tsao, "The SP1 high-performance switch," in *Proceedings of the Scalable High Performance Computing Conference*, pp. 150–157, 1994.

[14] C. B. Stunkel, D. G. Shea, and B. Abali, *et al*, "The SP2 high-performance switch," *IBM System Journal*, vol. 34, pp. 185–204, February 1995.

[15] H. Xu, Y. Gui, and L. M. Ni, "Optimal software multicast in wormhole routed multistage networks," in *Proceedings of Supercomputing*, pp. 703–712, 1994.

[16] C. Chiang and L. M. Ni, "Efficient software multicast in wormhole-routed unidirectional multistage networks," in *Proceedings of the Symposium of Parallel and Distributed Processing*, pp. 106–113, 1995.

[17] J. Park, H. Choi, N. Nupairoj, and L. Ni, "Construction of optimal multicast trees based on the parameterized communication model," in *Proceedings of the International Conference on Parallel Processing*, 1996.

[18] L. M. Ni, "Should scalable parallel computer support efficient hardware multicast?," in *Proceedings of the ICPP Workshop on Challenges for Parallel Processing*, pp. 2–7, 1995.

[19] D. K. Panda, "Issues in designing efficient and practical algorithms for collective communication on wormhole-routed systems," in *Proceedings of the ICPP Workshop on Challenges for Parallel Processing*, pp. 8–15, 1995.

[20] C. Chiang and L. M. Ni, "Deadlock-free multi-head wormhole routing," in *Proceedings of the First High Performance Computing-Asia*, 1995.

[21] D. K. Panda and R. Sivaram, "Fast broadcast and multicast in wormhole multistage networks with multidestination worms," Tech. Rep. OSU-CISRC-04/95-TR21, Department of Computer and Information Science, Ohio State University, 1995.

[22] M. P. Malumbres, J. Duato, and J. Torrellas, "An efficient implementation of tree-based multicast routing in distributed shared-memory multiprocessors," in *Proceedings of the Eighth Symposium on Parallel and Distributed Processing*, pp. 186–189, October 1996.

[23] R. Sivaram, D. K. Panda, and C. B. Stunkel, "Fast broadcast and multicast on wormhole multistage networks using multiport encoding," in *Proceedings of the Eighth IEEE Symposium on Parallel and Distributed Processing*, pp. 36–45, October 1996.

[24] L. M. Ni and P. K. McKinley, "A survey of wormhole routing techniques in direct networks," *IEEE Computer*, vol. 26, pp. 62–76, February 1993.

[25] W. J. Dally and C. L. Seitz, "Deadlock-free message routing in multiprocessor interconnection networks," *IEEE Transactions on Computers*, vol. 36, pp. 547–553, May 1987.

[26] C. B. Stunkel, R. Sivaram, and D. K. Panda, "Implementing multidestination worms in switch based parallel systems: Architectural alternatives and their impact," in *Proceedings of the International Symposium on Computer Architecture*, pp. 50–61, June 1997.

[27] S. Balakrishnan and D. K. Panda, "Impact of multiple consumption channels on wormhole routed k-ary n-cube networks," in *Proceedings of the International Parallel Processing Symposium*, pp. 163–167, 1993.

[28] H. J. Siegel, "The theory underlying the partitioning of permutation networks," *IEEE Transactions on Computers*, vol. C-29, pp. 791–800, September 1980.

[29] M. D. Schroeder, A. D. Birrell, M. Burrows, H. Murray, R. M. Needham, T. L. Rodeheffer, E. H. Satterthwaite, and C. P. Thacker, "Autonet: A high-speed self-configuring local area network using point-to-point links," tech. rep., SRC Research Report 59, 1990.

[30] K. V. Anjan and T. M. Pinkston, "*DISHA:* A deadlock recovery scheme for fully adaptive routing," in *Proceedings of the International Parallel Processing Symposium*, April 1995.

[31] H. D. Schwetman, "Introduction to process-oriented simulation and CSIM," in *Proceedings of the Winter Simulation Conference*, pp. 154–157, December 1990.