A Hardware Multicast Routing Algorithm for Two-Dimensional Meshes

Prasant Mohapatra and Vara Varavithya
Department of Electrical and Computer Engineering
Iowa State University, Ames, IA 50011
E-Mail: prasant@iastate.edu

Abstract

Multicast communication is a significant operation in multicomputer systems and can be used to support several other collective communication operations. Hardware implementation is a viable solution to develop a low latency multicast algorithm. In this paper, we present a new multicast routing algorithm for two-dimensional meshes. The algorithm uses wormhole routing mechanism and can send messages to any number of destinations within two start-up communication phases; hence the name, two-phase multicast (TPM) algorithm. The algorithm uses the base routing scheme used for unicast communication, thus minimizing the additional hardware support. The algorithm allows some intermediate nodes that are not in the destination set to perform multicast functions. This feature allows flexibility in multicast path selection and therefore improves the performance. A simulation study has been conducted to investigate the performance of the purposed TPM algorithm. The simulation results show that the proposed TPM algorithm performs significantly better for both contention-free and mixed-traffic conditions, when compared with the previously proposed multicast algorithms.

1 Introduction

In high performance parallel computers, processors communicate between each other by passing messages via the communication network. Thus a low latency communication mechanism is required to keep up with the processing speed. In most contemporary parallel systems, low dimensional wormhole-routed k-ary n-cube networks are adopted due to its low communication latency and high bandwidth [1]. The wormhole routing scheme used in most of these systems supports only unicast (one to one) communication.

Collective communication which involves a group of intercommunicating nodes is useful in several applications [2]. Some parallel programming languages provide efficient support for these applications. The importance of collective communication is further demonstrated by their inclusion in the message passing interface (MPI) standards [3]. Examples of collective operations include broadcast, multicast, gather, scatter, all-gather, barrier synchronization, and global reduction. Several of these operations can be supported through an efficient implementation of the multicast communication. Multicast communication is concerned with the delivery of a message from one source node to multiple destinations. The multicast services can be considered as basic services for other collective operations. It is also used for distributing data to processes. In shared memory systems, multicasting is used for cache invalidations.

The performance of multicast communications is measured in terms of its latency in delivering a message to all destinations. In wormhole routed networks, the communication latency consists of two parts, start-up latency and network latency. The start-up latency is the time required to start a message which involves operating system overheads. The network latency is a combination of propagation delay, router delay, and contention delay. The communication latency is dominated by the start-up latency which is in order of 1 μs to 20 μs in the current generation systems [4]. For example, consider a 16×16 mesh network having the following parameters: 3.2 Gbit/sec link bandwidth, 64 bits/flit, 1024 bits messages. Assume the average number of hops is equal to 14 using the uniform traffic pattern. The average network latency will be $(\frac{64}{(3.2 \times 10^9)}) \times (14+15) = 580ns$. If we assume $1\mu s$ start-up time, the start-up latency accounts for 63.29% of total communication latency. In multicast communications, where an operation might consist of several communication phases, the start-up latency has a significant impact on the performance.

Multicast communication can be carried out by sending one message to each destination. This method performs poorly not only because it requires a large amount of network resources but also involves too many communication steps. An efficient software multicasting technique called Umesh is proposed for meshes that do not require any hardware modification [5]. The Umesh scheme reduces the number of communication steps by allowing some destinations to act like source nodes after they receive the message. However, this technique still involves a large number of communication start-up steps and software overheads.

To further reduce the communication latency, the multicast operations need to be supported by the hardware. The importance of hardware supported multicasting was reported by Ni [6]. In [7], the Hamiltonian path based algorithms were proposed to reduce the amount of traffic created from the multicast operations. However, in this algorithm, the path length becomes a dominant factor, leading to high latency. Furthermore, the Hamiltonian path-based scheme does not conform to any base routing scheme proposed for unicast communications.

Column-path [8] and leader-based [9] multicast algorithms are two similar algorithms developed independently and conforms to the base routing algorithm. In [9], it was reported that the number of communication steps for single-source multicast for a destination set DS in a $k \times k$ mesh is less than or equal to $\lceil \log_t(2k) \rceil + 2$ if $t > 1$ and $\lceil \log_t(k) \rceil + 2$ if $t \leq 1$, where $kt \leq |D| \leq k(t+1)$ and t is an integer. We have attempted to further reduce the number of start-up steps and lower the latency. In this paper, we propose a new hardware multicast algorithm that conforms to the base routing scheme as well as uses at the most two communication start-up phases to reach any number of destinations in a two-dimensional (2-D) Mesh. The methodology is called two-phase multicast (TPM) algorithm. To fully utilize the multicast paths supplied by the base routing algorithm, some intermediate nodes that are not destinations are allowed to perform multicast operations. This feature increases flexibility in distributing messages to the destinations thereby improving performance. The performance of the multicast algorithm is evaluated through simulations. The simulation results show that the proposed algorithm performs better than the dualpath [7] and column-path [8] algorithms.

This paper is organized as follows. Section 2 presents preliminaries for the multicast models. The proposed algorithm is described in Section 3. A deterministic TPM algorithm based on the dimension order routing is reported in Section 4. The simulation results are presented in Section 5. Finally, concluding remarks are given in Section 6.

2 Preliminaries

The basic node architecture of a two-dimensional (2D) mesh is illustrated in Figure 1. The router provides communication services to the host processor. The incoming/outgoing internal channels to/from the router are usually referred as injection/consumption channels. The number of injection/consumption channels implies the number of messages that can be sent/received concurrently by the processor. The number of internal channels is sometimes used to classify the communication system architecture [2]. The one-port model refers to the system that has one pair of internal channels at each node such that a node can receive or send only one message at any instant of time. In an all-port model, the number of injection channels and consumption channels are equal to the number of input and output channels, respectively.

Hardware multicasting requires some additional functionality implemented in the router along with the abilities associated with unicast communication. Some basic communication services required in wormhole routed networks for multicasting are defined as follows.

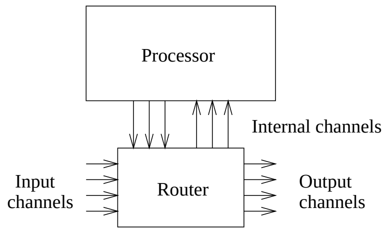


Figure 1: Router architecture.

- Forward (FWD): When the router accepts a message from a neighboring node and the current node is not one of the destination(s), the message is forwarded to the next node toward its destination(s).

- Absorb (ABS): If the current node is the destination, the message is absorbed and passed on to the local processor.

- Retransmit (RTM): For deadlock avoidance purpose, in some cases, the multicast message must be temporarily stored and retransmitted to eliminate the cyclic dependencies. The message is stored in the local memory. Thus the probability of deadlock from the depletion of storage space is negligible. The overhead of retransmission operation is very high because it involves both software (operating system) and hardware resources.

The absorb and forward capability introduces additional resource dependencies that can lead to deadlock situations. Since multiple consumption channels may be occupied by one message along the multicast path, the deadlock configuration can stem from the cyclic waiting for consumption channels. An example of consumption channel deadlock in a linear-array is shown in Figure 2. Node 1 generates message A destined to nodes 2 and 3. At the same time, message B is generated by node 4 destined to the same set of destinations. Assuming an one-port model, after two steps, message A occupies consumption channel $c2$ at node 2 and requests for consumption channel $c3$ at node 3. The consumption channel $c3$ is occupied by the message B which also requests for $c2$. This cyclic wait creates a deadlock. The deadlock due to consumption channel contention can involve several nodes. The upper bound of the number of consumption channels required to avoid such deadlocks is equal to nv where n is the network dimension and v is the number of virtual channels per direction [8]. The detailed treatments of the consumption channels deadlock can be found in [8, 10].

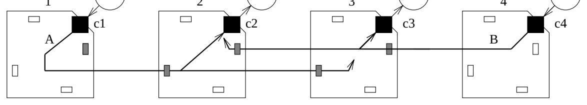


Figure 2: Deadlock due to consumption channels.

3 A Two-Phase Multicast Algorithm

In this section, we present a new hardware multicast routing algorithm that uses at most two start-up phases to communicate with any number of destinations; hence the name *two phase multicast* (TPM) algorithm.

3.1 Basic Characteristics

There are two essential characteristics of the proposed algorithm. First, the TPM algorithm uses the same routing algorithm as that employed for unicast communication and thereby does not introduces any deadlock (assuming that the base routing algorithm is deadlock-free). The second characteristic of the TPM algorithm differentiates it from all the previously proposed algorithms. In the previously proposed multicast algorithms, only the multicast destination nodes participate in the multicast operations. This restriction increases the number of communication phases, therefore degrades the performance. In the TPM scheme, we allow some additional nodes that are not in the destination set to perform multicast operations. This mechanism can be supported by introducing some additional communication primitives as itemized later in this section.

Because of the involvement of nodes that are not destinations, it may appear that the performance of these nodes will degrade. However, the reduction of multicast latency overcomes this degradation. Furthermore, as we involve nodes that are in the multicast zone (defined and discussed later), and as the tasks are allocated in forms of submeshes, the additional nodes that are forced to be involved in a multicast operation always belong to the same task. So the inclusion of these additional nodes for multicasting does not affect any other tasks. Above all, the TPM algorithm always tries to minimize the involvement of nodes that are not destinations. The performance improvement obtained using TPM algorithm justifies the temporary involvement of nodes that are not destinations.

3.2 Hardware Support

The multicast communication services used by the TPM algorithm are extended from the basic communication services (listed in the previous section) and are summarized as follows:

- Permanent absorb and forward (PAF): If an intermediate node is one of the multicast destinations, the message is absorbed in the node while being forwarded to the next node.

- Permanent absorb and retransmit (PAR): The intermediate node performs a PAR operation if it is the last node in the first phase of communication and is also one of the multicast destinations.

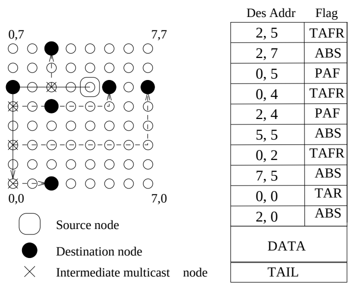
- Permanent absorb, forward, and retransmit ($PAFR$): For a two-phase multicast operation, some intermediate nodes need to absorb and retransmit a message to another set of destinations. If the intermediate node that performs such operation is also a multicast destination, the $PAFR$ operation is executed.

- Temporary absorb and retransmit (TAR): In some cases, we might need to retransmit at an intermediate node that is not one of the destinations. Such intermediate nodes will temporarily store a message and retransmit later in the next phase.

- Temporary absorb, forward, and retransmit ($TAFR$): TAFR is similar to TAR except that the message is also forwarded to the next node while it is being temporarily stored at the current node.

3.3 Destination Preprocessing and Multicast Zone

The destination addresses along with the nodes that will perform multicast operations are sorted in order and encapsulated in the message header. Along with the addresses, the required multicast services are also stored in the header in term of flags. An example of a message structure showing ten destinations is shown in Figure 3. The required multicast operations are also tagged along with the intermediate destination addresses. For example, the node $(0,4)$, which is not a destination, need to perform the TAFR operation to retransmit the multicast message to the node (2,4) and (5,5). The node (2,4) flag is tagged as PAF to perform absorb while forwarding to node (5,5). Since the node (5,5) is the last destination in the branch, the ABS operation is assigned.



Des Addr	Flag
2, 5	TAFR
2, 7	ABS
0, 5	PAF
0, 4	TAFR
2, 4	PAF
5, 5	ABS
0, 2	TAFR
7, 5	ABS
0, 0	TAR
2, 0	ABS
DATA	
TAIL	

Figure 3: Encapsulation of destination addresses in the header.

Multicast zone is defined as the smallest two-dimension array that includes the source node and all the destinations. The purpose of zoning is to confine a boundary of network resources that can be utilized for the multicast operations. Let the coordinates of the lower left corner be denoted by (l_x, l_y) and the upper right corner be denoted by (u_x, u_y), as depicted in Figure 4.

We logically divide the multicast zone into four quadrants. The quadrants are labeled as Q_1, Q_2, Q_3, and Q_4 as shown in Figure 4. The source quadrant is determined such that the farthest corner node with respect to the source node can be specified. The source quadrant can be obtained using the algorithm shown in Figure 5. To determine the source quadrant, the source coordinates are compared to the center of the multicast zone. The TPM algorithm uses different destination ordering functions based on the source

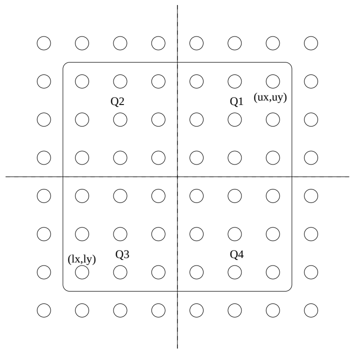


Figure 4: Quadrants in the multicasting zone.

Algorithm 1: Algorithm for determination of the source quadrant.
Input: Multicast zone (l_x, l_y, u_x, u_y), source node (s_x, s_y).
Output: The source quadrant SQ, relative positions of periphery lines (N_h, F_h, N_v, F_v).
Procedure:

```
Let $midx = u_x - l_x/2$; $midy = u_y - l_y/2$
begin
    if($S_x > midx$) and ($S_y > midy$)
        $SQ = 1, N_h = u_y, F_h = l_y, N_v = u_x, F_v = l_x$
    if($S_x < midx$) and ($S_y > midy$)
        $SQ = 2, N_h = u_y, F_h = l_y, N_v = l_x, F_v = u_x$
    if($S_x < midx$) and ($S_y < midy$)
        $SQ = 3, N_h = l_y, F_h = u_y, N_v = l_x, F_v = u_x$
    if($S_x > midx$) and ($S_y < midy$)
        $SQ = 4, N_h = l_y, F_h = u_y, N_v = u_x, F_v = l_x$
end.
```

Figure 5: Multicast source quadrant algorithm.

quadrant. The relative positions N_h, F_h, N_v, and F_v are defined for the ease of writing the destination preprocessing algorithm. The N_h, F_h, N_v, and F_v stand for near horizontal periphery line, far horizontal peripheral line, near vertical peripheral line, and far vertical peripheral line, respectively. These parameters are relative to the source coordinate and the multicast zone.

3.4 Framework of the TPM algorithm

The formal description of the routing algorithm is given in Figure 6. A message is forwarded to the next node if the current node is not a destination node. If the current node is a destination, the associated multicast_flag is used to determine the multicast operation to be performed. For the TPM algorithm, the implementation of three extra bits are adequate to represent all multicast operations that we have listed in Section 3.2. Each bit in the multicast flag represents a unique multicast service: absorb, forward, or retransmit. If the absorb flag is set, the message is copied to the application process. If the forward flag is set then the message is sent to the neighboring node in the direction of the next destination node. For retransmission, the destination list in the message header needs to be modified. The destination addresses starting from the next destination to first absorb-only destination are removed from the message header. The destination that performs only absorb operation implies the last destination in its branch. The destination addresses removed from the message header are passed on to the retransmission routing phase along with the data to be retransmitted.

Algorithm 2: Message routing operation.
Input: Current node (x_1, x_2), Message header(contain the destination set DS and the multicast flags in order of traversal).
Procedure:

```
begin
    1. if the current node address is the head of
         the destination in the header do
         multicast_operation(multicast_flag, DS)
    2. if the current node is not in the DS,
         forward the message to the next node using
         the underlying routing algorithm.
end.
```

Routine: Multicast_operation().
Input: multicast_flag, Message header(contain the destination set DS and the multicast flags in order of traversed).
Procedure:

```
begin
    1. remove the current destination from the message
         header.
    2. if the multicast_flag includes absorb operation
         -send copy of the received message to the application.
    3. if the multicast_flag includes retransmit operation
         -remove the destination addresses from the header
         (starting from the next destination in $DS$ to the
         first destination with only absorb operation).
         -pass the removed destination addresses and the
         content of the message to the retransmission routine.
    4. if the multicast_flag includes forward operation
         compute the next direction and forward the
         message to the next node.
end.
```

Figure 6: The TPM algorithm.

The basic idea behind the proposed algorithm is that, during the first step, the message is sent to a set of nodes such that all the destinations can be reached in the first or second phases of communication. First, the TPM algorithm defines the *multicast zone* that contains the source node and all the destinations. Next, it defines a path that is taken in the first step to the farthest corner (from the source node) of the multicast zone. This path is called the *main path*. After the first phase is completed, some of the intermediate nodes along the main path retransmit messages to rest of the multicast destinations. The main path is selected such that it conforms to the base routing algorithm and covers as many destinations as possible. This feature allows the multicast operations to be completed in at most two communication steps in 2-D meshes. As mentioned earlier, the communication latency in multicomputers is dominated by the start-up latency. Hence, minimization of the number of start-up steps is an important design issue for multicast algorithms.

4 Dimension-order TPM algorithm

In this section, we explain the detailed mechanism of the algorithm using a deterministic (dimension-order) routing scheme.

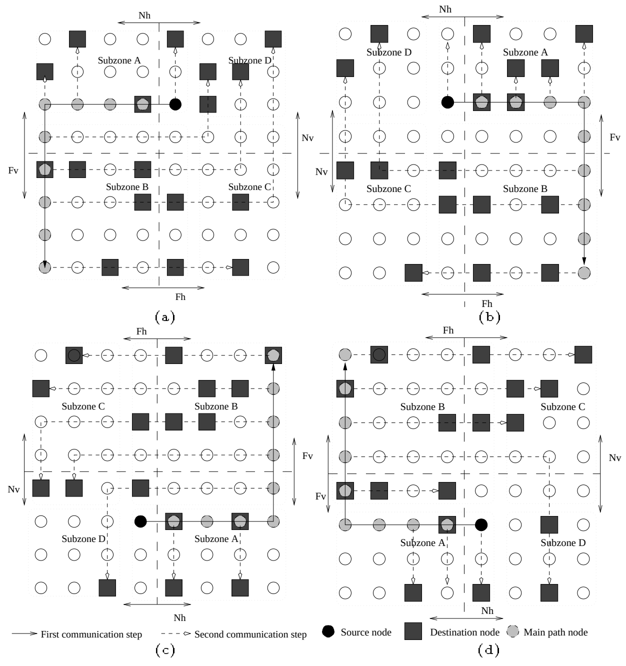


Figure 7: Deterministic TPM patterns (a) Source quadrant 1, (b) Source quadrant 2, (c) Source quadrant 3, (d) Source quadrant 4.

In the first step of multicast, the multicast message is sent to the farthest corner from the source node. The dimension order XY routing scheme [11] supplies a unique minimal path to the corner node of the diagonally opposite quadrant, hence there is no routing freedom to select the main path. The last destination node of the first multicast phase can be (l_x, l_y), (u_x, l_y), (u_x, u_y), or (l_x, u_y) corresponding to the source quadrant Q_1, Q_2, Q_3, or Q_4, respectively. In the second phase, some of the intermediate nodes along the main path retransmit messages to other multicast destinations that are not covered in the first phase. Figure 7 shows the TPM dimension order multicast patterns for XY routing on an (8×8) mesh network. The solid lines and dashed lines represent communication in the first and second steps, respectively. In the first step, the source node uses XY routing to travel to the farthest corner node, i.e., to the corner node of the diagonally opposite quadrant. In the second step, some of the nodes that received the message during the first step send the message to the remaining destinations.

After the farthest corner node has been reached using the main path, all other destination nodes are guaranteed to be covered by the intermediate nodes during the second phase. However, some shape restrictions need to be applied for the deterministic routing. With XY routing, the TPM algorithm cannot cover all destinations if the X dimension is larger than the Y dimension. To avoid such problem, the multicast is extended zone such that $|S_y - F_h| \geq |S_x - N_v|$. Thus, for rectangular meshes, XY routing is used if $X \leq Y$, else YX routing is employed for both unicast and multicast communications.

In the TPM destination preprocessing algorithm, the nodes in the main path are assigned all the multicast destinations to be covered in the second phase. The formal description of this operation is shown in Figure 8. The multicast destinations are divided into four subzones with respect to the source node, as shown in Figure 7. The algorithm in Figure 9 outlines the subzone labeling scheme with respect to the source node. To explain the algorithm, we use quadrant 1 as an example. For destination nodes in subzone A, $[(l_x \leq D_x \leq S_x), (S_y \leq D_y \leq u_y)]$, the corresponding main path nodes have coordinate: (D_x, S_y). Similarly, for destination nodes in subzone B, $[(l_x \leq D_x \leq S_x), (ly \leq D_y < S_y)]$, the corresponding main path nodes have coordinate (F_v, D_y), where F_v is equal to l_x. The main path nodes for destinations in subzone D can be obtained by mapping the addresses to the F_v column. Some subzone C destinations also need mapping if it overlaps with the subzone D routing paths.

Lemma 1: The deterministic TPM algorithm requires only two communication start-up steps for 2-D mesh networks.

Proof: The message is sent to the opposite diagonal corner of the multicast zone in the first phase. If the multicast zone is defined such that $|S_y - F - h| \geq |S_x - N_v|$, all other destinations in the multicast zone can be reached using the XY paths in the second phase.

Lemma 2: The deadlock-free TPM requires four consumption channels in 2-D mesh networks.

Proof: The TPM algorithm allows the absorb and forward operations to be perform in only X dimension, only Y dimension, and both XY dimension. In this configuration, at most four consumption channels are required at each node. The detail proof is discussed in [9].

5 Performance Evaluation

To investigate the effect of multicast routing algorithm, we have developed a flit level wormhole routing simulator. The simulation environment and the per-

Algorithm 3: TPM destinations ordering and header encapsulation.
Input: Destination set: $DS[(d_{0x}, d_{0y}), \ldots, (d_{nx}, d_{ny})]$, Multicast periphery: (l_x, l_y, u_x, u_y), source node: (S_x, S_y), relative position: (N_h, F_h, N_v, F_v).
Output: Encapsulated destination message header.
Procedure: Let $MS[(m_{0x}, m_{0y}), \ldots, (m_{nx}, m_{ny})]$ be the set of main path nodes associated with the DS. DM array contains destinations associated with the main path nodes, $DM < (d_{0x}, d_{0y}), (m_{0x}, m_{0y}) >, \ldots, < (d_{nx}, d_{ny}), (m_{nx}, m_{ny}) >]$.

```
begin
  1. for all elements in the $DS$
     if Destination_subzone($(d_{ix}, d_{iy})$) = subzone A
        $m_{ix} = d_{ix}$, $m_{iy} = S_y$
     if Destination_subzone($(d_{ix}, d_{iy})$) = subzone B
        $m_{ix} = F_v$, $m_{iy} = D_{iy}$
     if Destination_subzone($(d_{ix}, d_{iy})$) = subzone C
        $m_{ix} = F_v$
        if $| d_{iy} - S_y | < | N_v - S_x |$)
           if $N_h > S_y$, $m_{iy} = d_{iy} - max(| S_x - d_{ix} |,$
              $| S_y - d_{iy} |)$
           else $m_{iy} = d_{iy} + max(| S_x - d_{ix} |, | S_y - d_{iy} |)$
        else $m_{iy} = d_{iy}$
     if Destination_subzone($(d_{ix}, d_{iy})$) = subzone D
        $m_{ix} = F_v$
        if $N_h > S_y$, $m_{iy} = D_{iy} - | D_{ix} - S_x |$
        else $m_{iy} = D_{iy} + | D_{ix} - S_x |$
  2. Sort $DM = [< DS, MS >]$ according to main path
  in each quadrant using $MS$ and $DS$ as the sorting keys.
  3. Encapsulate the destination addresses along with
  the associated main path nodes that are not in
  the destination set into the header in order.
  4. Assign multicast_flags to the destination addresses.
end.
```

Figure 8: Header encapsulation process in deterministic TPM algorithm.

formance results are described in this section.

5.1 Simulation Model

A simulation model was developed for a 16 × 16 mesh. Fixed size packets of 20 flits per packet was assumed. To improve the traffic flow, we have considered two virtual channels per physical channel with one flit buffer per virtual channel. The time required to transfer one flit from a node to its neighbor is assumed to be equal to $30ns$ which is directly mapped to one time unit. The start-up latency is set to $1\mu s$ (33 time units). In multicast operation, multiple messages can be generated in the same communication phase. To model such situation, the succeeding message start-up time is set to $240ns$ (8 time units) because the generation of succeeding messages should take less time than the generation of the first message. If the multicast algorithm requires absorb and retransmit operation, the start-up latency is used to penalize the retransmission overhead for fair comparison.

Algorithm 4: Subzone labeling for each destinations.
Input: Destination: (d_{ix}, d_{iy}), Multicast periphery: (l_x, l_y, u_x, u_y), source node: (S_x, S_y), relative position: (N_h, F_h, N_v, F_v).
Output: Subzone label of the destination.
Procedure:

```
begin
  if( $S_x \le d_{ix} \le F_v$ or $S_x \ge d_{ix} \ge F_v$)
     if($S_y \le d_{iy} \le N_h$ or $S_y \ge d_{iy} \ge N_h$) return A
     if($S_y < d_{iy} \le F_h$ or $S_y > d_{iy} \ge F_h$) return B
  if( $S_x < d_{ix} \le N_v$ or $S_x > d_{ix} \ge N_v$)
     if($S_y < d_{iy} \le F_h$ or $S_y > d_{iy} \ge F_h$) return C
     if($S_y < d_{iy} \le F_h$ or $S_y \ge d_{iy} \ge F_h$) return D
end.
```

Figure 9: Labeling subzones with respect to the source node.

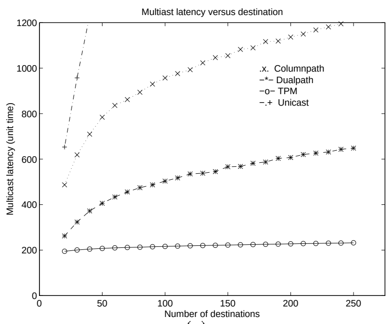
Some multicast operations require header processing in the intermediate nodes where an additional overhead needs to be taken into account. The routing decision usually take 1.5 to 2 times more than the time required for transferring the data flit [12]. Using this data, the routing decision is assumed to take $60ns$ (2 time units) without header modification. None of the earlier reported works have considered the overhead associated with header modification in multicast communications. We assumed $90ns$ (3 time units) routing delay for the routing decision and header modification.

5.2 Performance of Contention-Free Communication

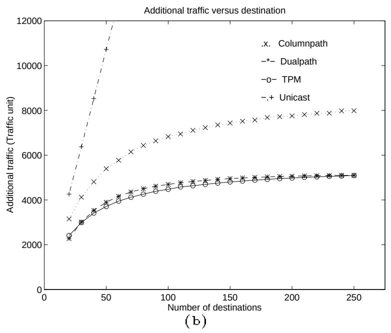
We first present the performance results under the contention-free condition. The source node is randomly chosen for each operation. The multicast destinations are assumed to be uniformly distributed over the network. The average performance metrics are calculated from 1000 multicast operations. The number of destinations is varied from 20 to 250 nodes. We have considered multicast latency and average additional traffic. as multicast performance metrics. The multicast latency is defined as the time between the initiation of multicast operation and the time when the tail of the multicast message reaches all the destinations. The additional traffic reflects the amount of network resources that are used to complete the multicast operation. A physical channel occupied for one time unit is counted as one traffic unit.

The performance of the proposed TPM algorithm is compared with the dualpath [7], column-path [8], and unicast-based multicast algorithms. In the unicast-based multicast algorithm, the source node sends one message to each destination in the multicast destination set.

We obtained simulation results for a one-port model where there is one consumption channel and one injection channel. Figure 10 (a) shows the multicast latency versus the number of destinations. The TPM algorithm has the lowest latency and is nearly constant with the increase in the number of destinations. The multicast latency for dualpath algorithm considerably increase when the number of destinations increase from 20 to 100 and thereafter flattens out. The column-path and the unicast based multicast perform poorly because they involve more number of communication steps when the number of destinations increase.



(a)

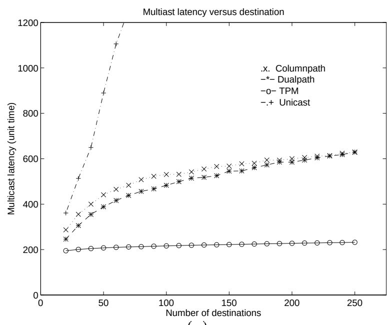


(b)

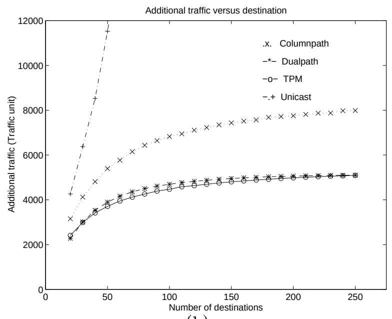
Figure 10: Performance comparison of multicast algorithms for an one-port model (a) Communication latency, (b) Additional traffic.

The results of additional traffic are shown in Figure 10 (b). The TPM algorithm requires a little less network resources than the dualpath algorithm. More network resources are required to complete the same multicast operation using unicast-based multicast or the column-path algorithms. This difference is attributed to the fact that, with the same number of destinations, the dualpath and TPM send the multicast messages in paths that cover more number of destinations.

In an all-port model, the number of consumption channels and the number of injection channels are equal to the total number of virtual channels in the router. The performance comparisons for an all-port model are shown in Figure 11. The trends are observed to be the same as that of the one-port model. However, the performance of the column path and unicast-based multicast performance are improved due to more number of messages being sent concurrently. The column-path algorithm performs as good as the dualpath algorithm. These results also show that the number of injection channels have a strong impact on the performance for the class of multicast algorithms that generate several messages at the same time.



(a)



(b)

Figure 11: Performance comparison of multicast algorithms for an all-port model (a) Communication latency, (b) Additional traffic.

5.3 Performance of Unicast and Multicast Traffic with Contention

In real applications, the multicast message might compete for network resources with the unicast messages or other multicast messages in the network. To examine the performance of the TPM algorithms in such situations, we have simulated the multicast traffic together with the unicast traffic. We consider 90 percent unicast messages and 10 percent multicast messages. The destinations are assumed to be uniformly distributed. The average number of destinations (D) in the multicast operation is set to 32 with standard deviation of 15. The minimum and maximum number of destinations is 2 and 250, respectively. As described in [8], the unicast inter-arrival time for each node is equal to $\frac{N}{X \times 0.9}$ where N is the number of nodes in the system and X is the system throughput. The multicast inter-arrival time is defined as $\frac{(N \times D)}{(X \times 0.1)}$. Our model assumed one injection channel and eight consumption channels implemented in the router.

The performance results under mixed traffic(unicast and multicast) are given in Figure 12. The unicast communication latency are shown in Figure 12 (a). Among all algorithms, the TPM algorithm give the best performance followed by the column-path, unicast-based multicast, and dualpath algorithms, respectively. The multicast latency curves are shown

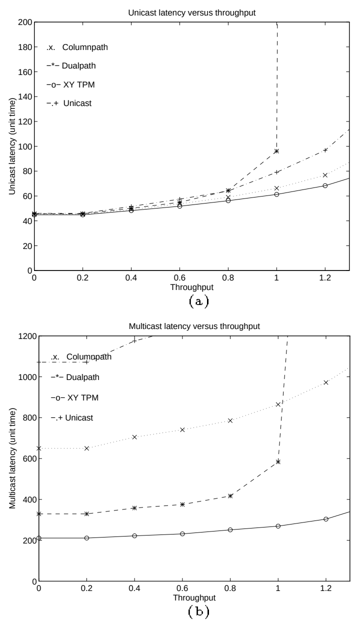


Figure 12: Performance results for mixed uniform and multicast traffic a) Unicast communication latency, b) Multicast communication latency.

in Figure 12 (b). The TPM algorithm has the lowest latency. The dualpath algorithm performs well in the low throughput but the latency increases rapidly in higher throughput. The communication latency increase slowly for column-path but it is severely degraded by the start-up latency and queuing delay at the source node.

6 Concluding Remarks

We have developed a new algorithm called two-phase multicast (TPM) algorithm for multicasting in two-dimensional mesh networks. The algorithm is very simple and requires at most two communication start-up steps to multicast to any member of destinations. During the first step, the message is sent to a node using a path such that the nodes covered during the first step can send the message to the remaining destinations in the second step, if required. The path taken in the first step guarantees that all the nodes will be reached within two start-up steps. 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 TPM algorithm performs better than the previously proposed multicast routing algorithms. Our future work is focused on extending the TPM algorithm for higher dimensional meshes.

Acknowledgement

This research was supported in part by the National Science Foundation under the grant MIP-9628801.

References

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

[2] 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.

[3] Message passing interface forum, *MPI: Message passing interface forum*, May 1994.

[4] Y. Tseng, D. K. Panda, and T. Lai, "A trip-based multicasting models in wormhole-routed networks with virtual channels," *IEEE Transactions on Parallel and Distributed Systems*, vol. 7, pp. 138–150, February 1996.

[5] 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.

[6] L. M. Ni, "Should scalable parallel computer support efficient hardware multicast?," in *Proceeding of the ICPP Workshop on Challenges for Parallel Processing*, 1995.

[7] 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.

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

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

[10] J. Duato, "A theory of deadlock-free adaptive multicast routing in wormhole networks," *IEEE Transactions on Parallel and Distributed Systems*, pp. 976–987, September 1995.

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

[12] W. J. Dally, R. Dennison, D. Harris, K. Kan, and T. Xanthopoulos, "The reliable router: A reliable and high-performance communication substrate for parallel computers," in *Processing of the Parallel Computer Routing and Communication Workshop*, pp. 241–255, May 1994.