Balancing Traffic Load for Multi-Node Multicast in a Wormhole 2-D Torus/Mesh

SAN-YUAN WANG[1], YU-CHEE TSENG[2], CHING-SUNG SHIU[3] AND JANG-PING SHEU[3]

[1]*Department of Information Engineering, I-Shou University, Kaohsiung, 840, Taiwan*
[2]*Department of Computer Science and Information Engineering, National Chiao-Tung University, Hsin-Chu, 300, Taiwan*
[3]*Department of Computer Science and Information Engineering, National Central University, Chung-Li, 320, Taiwan*
Email: sywang@isu.edu.tw

This paper considers the *multi-node multicast* problem in a wormhole-routed 2-D torus/mesh, where an arbitrary number of source nodes each intends to multicast a message to an arbitrary set of destinations. To resolve the contention and the congestion problems, we propose to partition the network into *subnetworks* to distribute, and thus balance, the traffic load among all network links. Several ways to partition the network are explored. The network-partitioning idea was used in earlier works for single-node broadcast and single-node multicast. This paper contributes in extending its applicability to multi-node multicast and demonstrating its capability to balance load on torus/mesh. Simulation results show significant improvement over existing results for torus and mesh networks.

Received 10 January 2000; revised 8 March 2001

1. INTRODUCTION

In a multicomputer network, processors often need to communicate with each other for various reasons, such as data exchange and event synchronization. Efficient communication is critical for high-performance computing. This is especially true for those collective communication patterns, such as *broadcast* and *multicast*, which involve more than one source and/or destination. Applications of collective communication include parallel numerical algorithms, graph algorithms, image processing algorithms, cache coherence and barrier synchronization [1, 2, 3, 4]. Some collective communications have also been implemented on standard communication libraries such as message-passing interface (MPI) [5] and collective communication library (CCL) [6].

Wormhole routing [7, 8] is becoming a mature switching technology for interconnection networks. It is characterized with low communication latency and is quite insensitive to routing distance in the absence of link contention. Such technology has been adopted by the Intel Touchstone DELTA [9], Intel Paragon [10, 11], MIT J-machine [12], Caltech MOSAIC [13], nCUBE 3 [14], Cray T3D and T3E [15, 16] and Myrinet [17].

This paper considers the *multi-node multicast* problem in a 2-D torus/mesh with wormhole, dimension-ordered and one-port routing capability. There are an arbitrary number of source nodes each intending to send a multicast message to an arbitrary set of destination nodes. We approach this problem by using multiple unicasts to implement multicast. The challenge is that there may exist serious contention when the source set or destination set is large or when there exist hot-spot effects (i.e. sources and/or destinations concentrated in some particular area). To resolve the contention problem, we apply two schemes, *network partitioning* and *load balancing*. We first partition the network into a number of 'subnetworks' and then evenly distribute these multicasts, by rerouting them to these subnetworks with the expectation of balancing the traffic load among all network links.

In the literature, more attention is on the single-node multicast problem [18, 19, 20, 21]. The work in [19] suggests connecting destination nodes as a *Hamiltonian path*, on which the multicast message can be sent in a pipelined manner. A widely used approach is the U-torus (respectively U-mesh) scheme by [20] (respectively [21]), in which destination nodes are connected as a *binomial tree*. With only one multicast, routing is guaranteed to be congestion free. The idea is further generalized by [18], where a Fibonacci tree is used instead. Works for multi-node multicast include [22] (for meshes with dimension-ordered routing) and [23, 24, 25] (for tori/meshes with non-dimension-ordered routing and multi-destination routing

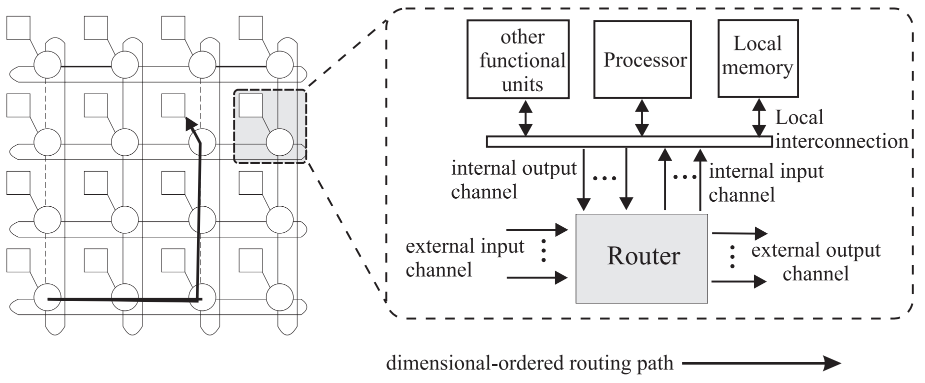


FIGURE 1. A 4 × 4 torus and the generic node architecture for wormhole routing.

capability). The work in [22] shows how to modify the U-mesh scheme to relieve the contention problem when there are multiple multicasts.

Our work is not to propose a completely brand-new scheme, in the sense that after a torus/mesh is partitioned, the obtained subnetworks are each a 'dilated' network still maintaining a similar torus/mesh topology. Thus, it is possible to apply the best available multicast schemes on these subnetworks. The details are in Section 2, where several ways to partition the torus/mesh are proposed. It is worth noting that the network-partitioning idea was originally proposed by the same authors in [26] and [27] for single-node broadcast and single-node multicast, respectively. The contribution of this paper is in extending its applicability to multi-node multicast, demonstrating its capability to balance load, and exploring more ways to partition a torus/mesh. Through extensive simulations, we justify that our network-partitioning approach can achieve better load balance and reduce multicast latency. Significant improvement can be obtained over the U-mesh/U-torus style of schemes [20, 21, 22] (which also assume dimension-ordered routing), especially when the traffic load is heavy or when there exists hot-spot behavior.

The rest of this paper is organized as follows. Preliminaries are in Section 2. Ways to partition a torus/mesh are presented in Section 3. Routing algorithms and simulations results on tori and meshes are in Section 4 and Section 5, respectively. Conclusions are drawn in Section 6.

2. PRELIMINARIES

This section contains a variety of information that serves as backgrounds for the materials to be presented later. Section 2.1 describes our network model. Section 2.2 describes the concept of network partitioning, where the definitions and properties of subnetworks, for wormhole networks in particular, are given. Without knowing how to partition torus and mesh, Section 2.3 gives a general three-phase multi-node multicast scheme based on subnetworks. This general scheme will later be fitted into particular partitioning methods and networks in subsequent sections.

2.1. Network model

A wormhole-routed multi-computer network consists of a number of computers (nodes) each with a separate *router* to handle its communication tasks [8]. From the connectivity between routers, we can define the topology of a wormhole-routed network as a graph $G = (V, C)$, where V is the node set and C specifies the channel connectivity. We assume the *one-port model*, where a node can send, and simultaneously receive, one message at a time. The opposite is the *all-port model*, where a node can send and receive messages via all its ports simultaneously. A generic architecture of a wormhole router and a torus connected by routers are shown in Figure 1. In this paper, we will consider networks that are connected as torus or mesh with *dimensional-ordered* routing.

In wormhole routing, a message is partitioned into a number of *flits* to be sent in the network. The *header* flit governs the routing, while the remaining flits simply follow the header in a pipelined fashion. In the contention-free case, the communication latency for sending a message of L bytes is commonly modeled by $T_s + LT_c$ [8], where T_s is the *startup time* (for initializing the communication) and T_c is the *transmission time* per byte. In this paper, different combinations of these parameters will be used for performance comparison.

2.2. Subnetworks of a wormhole network

DEFINITION 1. *Given a wormhole network* $G = (V, C)$, *a* subnetwork $G' = (V', C')$ *of* G *is one such that* $V' \subseteq V$ *and* $C' \subseteq C$.

For instance, Figure 2 shows four subnetworks, G_i, $i = 0 \ldots 3$, in a 16 × 16 torus. There are some subtleties in the above definition that need special attention.

- A subnetwork is not necessarily a 'graph' in standard graph theory. Specifically, suppose channel $(x, y) \in C'$. Then the vertices x and y are not necessarily in the vertex set V'. For instance, in Figure 2, the subnetwork G_0 contains links $(p_{0,0}, p_{0,1})$ and $(p_{0,1}, p_{0,2})$. However, only node $p_{0,0}$ is in G_0's node set.

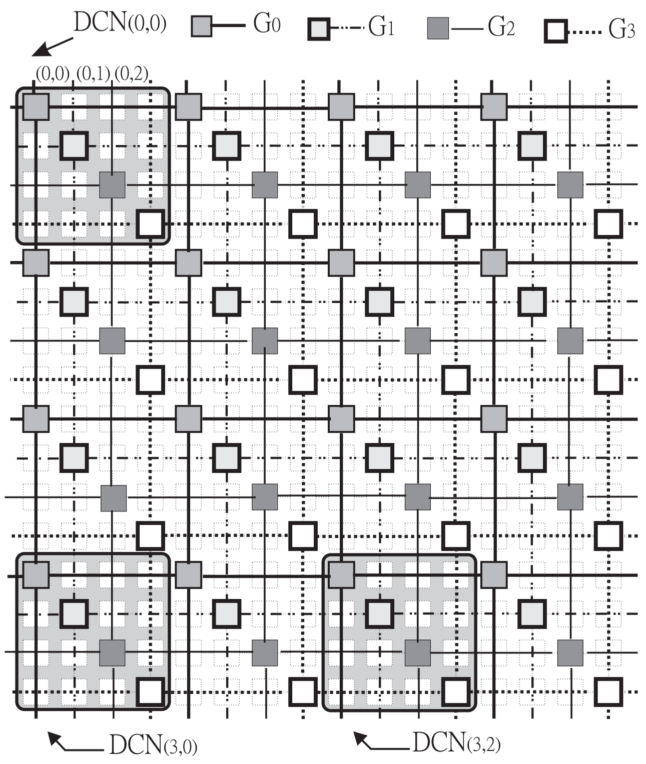


FIGURE 2. Four dilated-4 subnetworks, each as an undirected 4×4 torus, in a 16×16 torus. (For clarity, the wrap-around links on the boundaries are not shown.)

- The previous point carries special meanings for wormhole routing. For instance, each G_i in Figure 2 can be considered as a 4×4 torus, with each link 'dilated' by four links. However, the dilated torus can work almost like an ordinary torus, since communication in wormhole routing is known to be quite distance insensitive.
- A subnetwork, though capable of using all links in its link set, should be constrained in its capability in initiating/retrieving packets into/from the subnetwork subject to its node set. For instance, in Figure 2, nodes $p_{0,1}$ and $p_{0,2}$ are neither allowed to initiate a new worm into, nor retrieve a pass-by worm from, the subnetwork G_0. They can only passively relay worms they receive according to the routing function.

Our approach in this paper is to use multiple subnetworks in a torus to balance the communication load in different parts of the torus, thus eliminating congestion and hot-spot effects. This is of importance particularly for massive communication problems such as multi-node multicast. This leads to an important issue of making each subnetwork less dependent on other subnetworks, as formulated in the following definition.

DEFINITION 2. *Given two subnetworks $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, G_1 and G_2 are said to be* node-contention-free *if $V_1 \cap V_2 = \emptyset$, and* link-contention-free *if $E_1 \cap E_2 = \emptyset$.*

Intuitively, the freedom of node contention implies that we can freely schedule communications in each subnetwork without worrying that a node has to simultaneously initiate (and similarly, retrieve) worms for the two subnetworks. Similarly, freedom of link contention implies that two worms from different subnetworks will never contend with each other on the same link.

In cases where subnetworks are not fully disjoint, we use the following definition to identify the levels of sharing among nodes and links.

DEFINITION 3. *Given a set of subnetworks $G_1, G_2, \ldots, G_k$, the* level of node contention *(respectively* level of link contention*) among these subnetworks is defined to be the maximum number of times that a node (respectively, link) appears in these subnetworks, among all nodes (respectively, links) in the network.*

2.3. A general model for multi-node multicasts

A multi-node multicast instance can be denoted by a set of 3-tuple $\{(s_i, M_i, D_i), i = 1 \ldots m\}$. There are m source nodes $s_1, s_2, \ldots, s_m$. Each $s_i, i = 1 \ldots m$, intends to multicast a message M_i to a set D_i of destinations. Note that the destination sets need not be equal. The goal is to complete these multicasts as soon as possible.

Next, we derive a general approach to multi-node multicast based on the concept of subnetworks. Given any network G, we construct two kinds of subnetworks from G: *data-distributing networks* (DDNs) and *data-collecting networks* (DCNs). Suppose we have α DDNs, $DDN_0, DDN_1, \ldots, DDN_{\alpha-1}$ and β DCNs, $DCN_0, DCN_1, \ldots, DCN_{\beta-1}$. We require the following properties in our model.

P1 $DDN_0, DDN_1, \ldots, DDN_{\alpha-1}$ together incur on each node about the same level of node contention and similarly on each link about the same level of link contention.

P2 $DCN_0, DCN_1, \ldots, DCN_{\beta-1}$ are disjoint and they together contain all nodes of G.

P3 DDN_i and DCN_j intersect by at least one node, for all $0 \le i < \alpha$ and $0 \le j < \beta$.

Now given a problem instance $\{(s_i, M_i, D_i), i = 1 \ldots m\}$, a general approach is derived as follows.

Phase 1. Each multicast (s_i, M_i, D_i), $i = 1 \ldots m$, selects a target data distribution network, say, DDN_a to distribute its message. The selection should be done with load balance in mind. Then s_i chooses a node $r_i \in DDN_a$ as a representative of s_i in DDN_a and sends M_i to r_i.

Phase 2. From node r_i, perform a multicast (r_i, M_i, D'_i) on DDN_a, where the destination set D'_i is obtained from D_i by the following transformation. For each DCN_b, $b = 0 \ldots \beta - 1$, if DCN_b contains one or more destination nodes in D_i, select any node $d \in DDN_a \cap DCN_b$ (by **P3**) as the representative of the recipients of message M_i in DCN_b. Then we join d into D'_i.

Phase 3. In each DCN_b, $b = 0 \ldots \beta - 1$, after the representative node d receives M_i, it performs another multicast $(d, M_i, D_i \cap DCN_b)$ on the subnetwork DCN_b.

Intuitively, DDNs serve as the backbone to distribute multicast messages around the network, while DCNs will collect multicast messages for further forwarding. Phase 1 uses property **P1** to achieve load balance. Phases 2 and 3 are still a multicast, but they are on subnetworks DDN and DCN, respectively. So the model is in some sense a recursive one. Note that how to perform multicast in Phases 2 and 3 is left unspecified here.

The following two properties are not a necessity, but would offer regularity in designing phases 2 and 3.

P4 $DDN_0, DDN_1, \ldots, DDN_{\alpha-1}$ are isomorphic.

P5 $DCN_0, DCN_1, \ldots, DCN_{\beta-1}$ are isomorphic.

In the next section, we will discuss how to define the DDNs and DCNs in tori and meshes that satisfy our needs.

3. SUBNETWORKS OF A 2-D TORUS/MESH

We first discuss how to find DDNs and DCNs in a torus. Then we briefly summarize how to modify the definitions for a mesh.

3.1. *DDN*s and *DCN*s in a 2-D torus

A 2-D torus $T_{s\times t}$ consists of $s \times t$ nodes each denoted as $p_{x,y}$, where $0 \le x < s$ and $0 \le y < t$. Node $p_{x,y}$ has a link connected to each of $p_{(x\pm1)\bmod s,y}$ and $p_{x,(y\pm1)\bmod t}$. In the following, Definition 4 to Definition 7, we give four possible ways to define DDNs. Then in Definition 8 we will give a definition of DCNs to be used by all DDNs.

DEFINITION 4. *Given a torus $T_{s\times t}$ and any integer h that divides both s and t, define h subnetworks $G_i = (V_i, E_i)$, $i = 0 \ldots h-1$, such that*

$$V_i = \Big\{ p_{x,y} | x = ah + i,\ y = bh + i,\ \textit{for all } a = 0 \ldots \frac{s}{h} - 1 \textit{ and } b = 0 \ldots \frac{t}{h} - 1 \Big\}$$
$$C_i = \{\textit{all channels at rows } ah + i \textit{ and at columns } bh + i\}.$$

Intuitively, G_0 contains all nodes at the intersection of rows ah and columns bh, and G_i is obtained from G_0 by shifting G_0's nodes by i positions on both indices. In our terminology, each subnetwork is a 'dilated-h' torus of size $(s/h) \times (t/h)$. Figure 2 shows an example, with four subnetworks (each as a dilated-4 4×4 torus) in a 16×16 torus.

LEMMA 1. *The subnetworks G_i, $i = 0 \ldots h-1$, defined in Definition 4 are free from both node and link contention.*

Observe that in Definition 4, all links in the original torus have been used, so it is impossible to add more subnetworks without increasing link contention. However, there are still some nodes (e.g. nodes $p_{1,0}$ and $p_{0,1}$) that are not included in any subnetwork. The following definition tries to utilize these unused nodes to add more subnetworks without increasing node contention.

DEFINITION 5. *Given a torus $T_{s\times t}$ and any integer h that divides both s and t, define h^2 subnetwork $G_{i,j} = (V_{i,j}, E_{i,j})$, $i, j = 0 \ldots h-1$, such that*

$$V_{i,j} = \Big\{ p_{x,y} | x = ah + i,\ y = bh + j,\ \textit{for all } a = 0 \ldots \frac{s}{h} - 1 \textit{ and } b = 0 \ldots \frac{t}{h} - 1 \Big\}$$
$$C_{i,j} = \{\textit{all channels at rows } ah + i \textit{ and at columns } bh + j\}.$$

LEMMA 2. *The h^2 subnetworks $G_{i,j}$, $i, j = 0 \ldots h-1$, defined in Definition 5 are free from node contention, but have link contention of h.*

In the above definition, every node and every link have been used by some subnetwork(s), so it is impossible to add more subnetworks without increasing node and link contentions. However, we have only considered subnetworks with *undirected* links. With duplex capability, an undirected link can be regarded as two *directed* links in opposite directions. If we allow such separation, further improvement is possible. Let us call a direct link a *positive* link if it goes from a lower index to a higher one, and a *negative* link otherwise. The following is an extension of Definition 4.

DEFINITION 6. *Given a torus $T_{s\times t}$ and any integer h that divides both s and t, define h subnetworks $G_i^+ = (V_i^+, E_i^+)$, $i = 0 \ldots h-1$, such that (refer to Definition 4)*

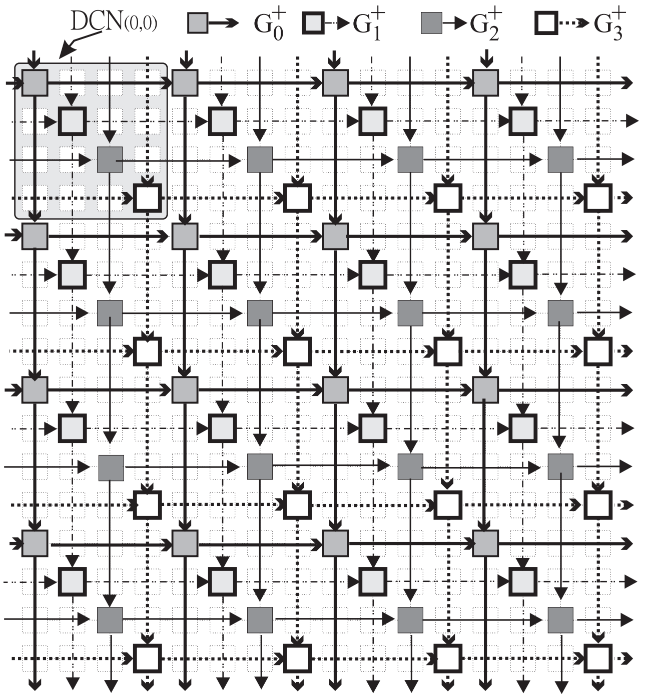
$$V_i^+ = V_i$$
$$C_i^+ = \{\textit{all positive links in } C_i\},$$

and h subnetworks $G_i^- = (V_i^-, E_i^-)$, $i = 0 \ldots h-1$, such that

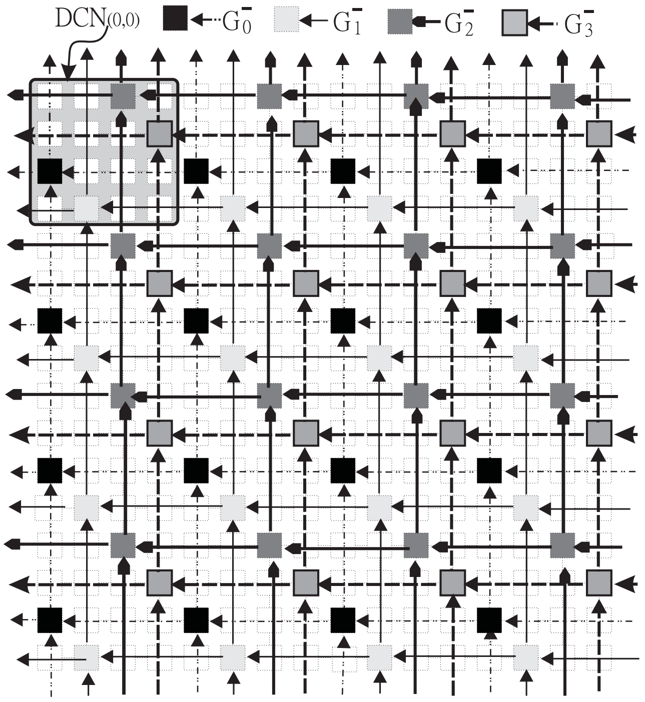
$$V_i^- = \Big\{ p_{x,y} | x = ah + i,\ y = bh + i + \delta,\ \textit{for all } a = 0 \ldots \frac{s}{h} - 1 \textit{ and } b = 0 \ldots \frac{t}{h} - 1 \Big\}$$
$$C_i^- = \{\textit{all negative links at rows } ah + i \textit{ and at columns } bh + i + \delta\},$$

where δ is any constant satisfying $1 \le \delta \le h - 1$.

Intuitively, G_i^+ is the same as G_i except that G_i^+ contains only positive links. Subnetwork G_i^- is obtained from G_i^+ by shifting each of the latter's nodes along the second dimension by δ positions and using only negative links. This is to resolve the node contention. For instance, Figure 3



(a)



(b)

FIGURE 3. Eight dilated-4 Type III subnetworks, each as a directed 4×4 torus in a 16×16 torus. (a) Type III subnetworks G_0^+, G_1^+, G_2^+ and G_3^+; (b) Type III (with $\delta = 2$) subnetworks G_0^-, G_1^-, G_2^- and G_3^-.

illustrates this definition in a 16×16 torus with $h = 4$ and $\delta = 2$ (for clarity, the eight subnetworks are drawn separately according to their link directions). Note that in the above definition, although each link is separated into a positive link and a negative link, subnetworks remain connected, in that each node in a subnetwork remains reachable by all other nodes in the subnetwork. Otherwise, routing would be difficult.

LEMMA 3. *The $2h$ subnetworks G_i^+ and G_i^-, $i = 0 \dots h-1$, defined in Definition 6, are free from both node and link contention.*

The following is an extension of Definition 5.

DEFINITION 7. *Given a torus $T_{s\times t}$ and any integer h that divides both s and t, define h^2 subnetworks $G_{i,j}^* = (V_{i,j}^*, E_{i,j}^*)$, $i, j = 0 \dots h-1$, such that (refer to Definition 5):*

$$V_{i,j}^* = V_{i,j}$$

$$C_{i,j}^* = \begin{cases} \{\textit{all positive links of } C_{i,j}\}, & \textit{if } i+j \textit{ is even}, \\ \{\textit{all negative links of } C_{i,j}\}, & \textit{if } i+j \textit{ is odd}. \end{cases}$$

LEMMA 4. *The h^2 subnetworks $G_{i,j}^*$, $i, j = 0 \dots h-1$, defined in Definition 7, are free from node contention, but have a link contention of $h/2$.*

In Table 1, we summarize the above definitions on the levels of node and link contention incurred by different subnetworks. In all cases, there is no node contention. One interesting property is that the level of link contention is proportional to the square root of the number of subnetworks. Also, the maximum number of subnetworks that can be obtained without incurring node and link contention is $2h$, given by Definition 7.

DEFINITION 8. *Given a torus $T_{s\times t}$ and any integer h that divides both s and t, define st/h^2 data collecting networks $DCN_{a,b} = (V_{a,b}, C_{a,b})$, $a = 0 \dots s/h-1$, $b = 0 \dots t/h-1$, such that*

$$V_{a,b} = \{p_{x,y} | x = a \times h + i, y = b \times h + j \\ \textit{for all } i, j = 0 \dots h-1\}$$
$$C_{a,b} = \{\textit{all (undirected) links induced by } V_{a,b}\}.$$

Intuitively, we simply partition the torus into a number of submeshes, each of size $h \times h$. For instance, when $h = 4$, Figure 2 illustrates the 16 DCNs (each as a 4×4 block) in a 16×16 torus. The same DCN definition will be used on all earlier four DDN definitions. Finally, it is not hard to see that these definitions satisfy properties **P1–P5**.

3.2. *DDN*s and *DCN*s in a 2-D mesh

A 2-D mesh $M_{s\times t}$ is similar to a torus $T_{s\times t}$ except that there are no 'wrap-around' links. The earlier four DDN definitions for torus can be easily translated to the mesh case by excluding the wrap-around links. However, reachability will become a problem when subnetworks are directed, as some nodes cannot reach all other nodes. This is in fact a main difference between a torus and a mesh. So we only consider using Definition 4 and Definition 5 to define our

TABLE 1. Comparison of levels of node and link contention incurred by different definitions of subnetworks in a torus.

Type	Subnet.	No. of subnet.	Links	Node cont.	Link cont.
I	$G_i, i = 0 \dots h-1$	h	Undirected	No	No
II	$G_{i,j}, i, j = 0 \dots h-1$	h^2	Undirected	No	h
III	$G_i^+, G_i^-, i = 0 \dots h-1$	$2h$	Directed	No	No
IV	$G_{i,j}^*, i, j = 0 \dots h-1$	h^2	Directed	No	$h/2$

DDNs. The levels of node and link contention remain the same. Also, the same DCNs in Definition 8 are used.

4. MULTI-NODE MULTICAST IN A 2-D TORUS

Given a multi-node multicast instance $\{(s_i, M_i, D_i), i = 1 \dots m\}$, next we show in more detail how to apply the multi-node multicast model of Section 2.3 using the DDNs and DCNs defined above. Throughout this section, let $DDN_0, DDN_1, \dots, DDN_{\alpha-1}$ be α DDNs obtained from Definition 4, 5, 6 or 7, and $DCN_0, DCN_1, \dots, DCN_{\beta-1}$ be β DCNs obtained from Definition 8.

4.1. Phase 1: balancing traffic among DDNs

In this phase, each multicast (s_i, M_i, D_i), $i = 1 \dots m$, should be distributed to one of the DDNs. There are two concerns to distribute the load. First, each DDN should receive about the same number of multicasts. Second, the overhead to achieve load balance should be as low as possible.

There are two ways to solve this problem. The first one is a centralized approach, where it is assumed that the multicast pattern is known in advance. Recall that we need to send each multicast to one target DDN. We can sequentially process DCNs one-by-one by looking at the multicasts on it. Specifically, for each DCN, we collect all multicasts (s_i, M_i, D_i) such that s_i is on the DCN. We assign these multicasts to those nodes that belong to a DDN in this DCN in a round-robin manner. Also, when doing such assignment, we always start from those DDNs that receive less multicasts in processing the previous DCN. In this way, only local communication will occur inside this DCN. Also, global load balance is achieved easily.

The second one is a distributed approach. We can simply let each s_i randomly choose one DDN as its target subnetwork. Since this is by randomization, the probability of achieving load balance might be high. This approach is more appropriate when the arrival of multicasts is unpredictable or follows a *stochastic* model, such as that assumed in [28]. One special case is when subnetworks of Types II or IV are used, where each node must belong to some subnetwork. It is possible to skip this phase by letting s_i serve as its own representative node. Load balance might be achieved automatically if sources are already distributed uniformly enough.

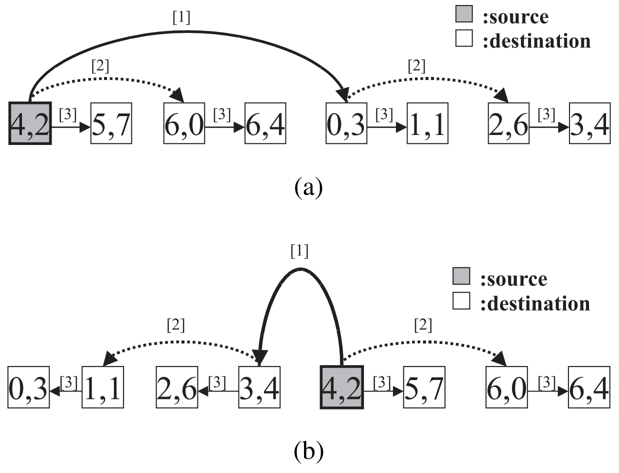


FIGURE 4. Single-node multicast. (a) The U-torus scheme, and (b) the U-mesh scheme.

4.2. Phase 2: multicasting in DDNs

In this phase, each multicast (s_i, M_i, D_i) is translated into a (r_i, M_i, D_i') to be performed in a DDN. Since each DDN is still a torus under our definition (except that there is some link dilation), this is still a multicast on a conceptually smaller torus (due to the distance-insensitive characteristic of wormhole routing). Also, it should be commented that the way that D_i is translated to D_i' will incur a concentration effect and thus there is a high probability that $|D_i'| < |D_i|$. So, the multicast is on a smaller network with a smaller destination set. Statistically, we can say that $|D_i'| \approx |D_i|/\alpha$.

Overall, each DDN will still need to perform a multi-node multicast. Improvement is expected if Phase 1 can deliver some load-balancing effect. With the dimension-ordered routing constraint, one possibility is to use the U-torus scheme [21] for each multicast. For completeness, we review the U-torus scheme below (using multicast (r_i, M_i, D_i') as an instance).

1. Consider the node set $\{r_i, D_i'\}$. We first sort the nodes in the set in an ascending order as follows. Nodes in a torus are totally ordered according to their indices such that $p_{i,j} < p_{i',j'}$ if $i < i'$ or $i = i'$ but $j < j'$.
2. Let the sorted list be $x_0, x_1, \dots, x_n$, where $n = 1 + |D_i'|$. Also let $x_a = r_i$. Then we 'left-rotate' the list into $Seq = x_a, x_{a+1}, \dots, x_n, x_0, x_1, \dots, x_{a-1}$ (i.e. r_i becomes the leading element).

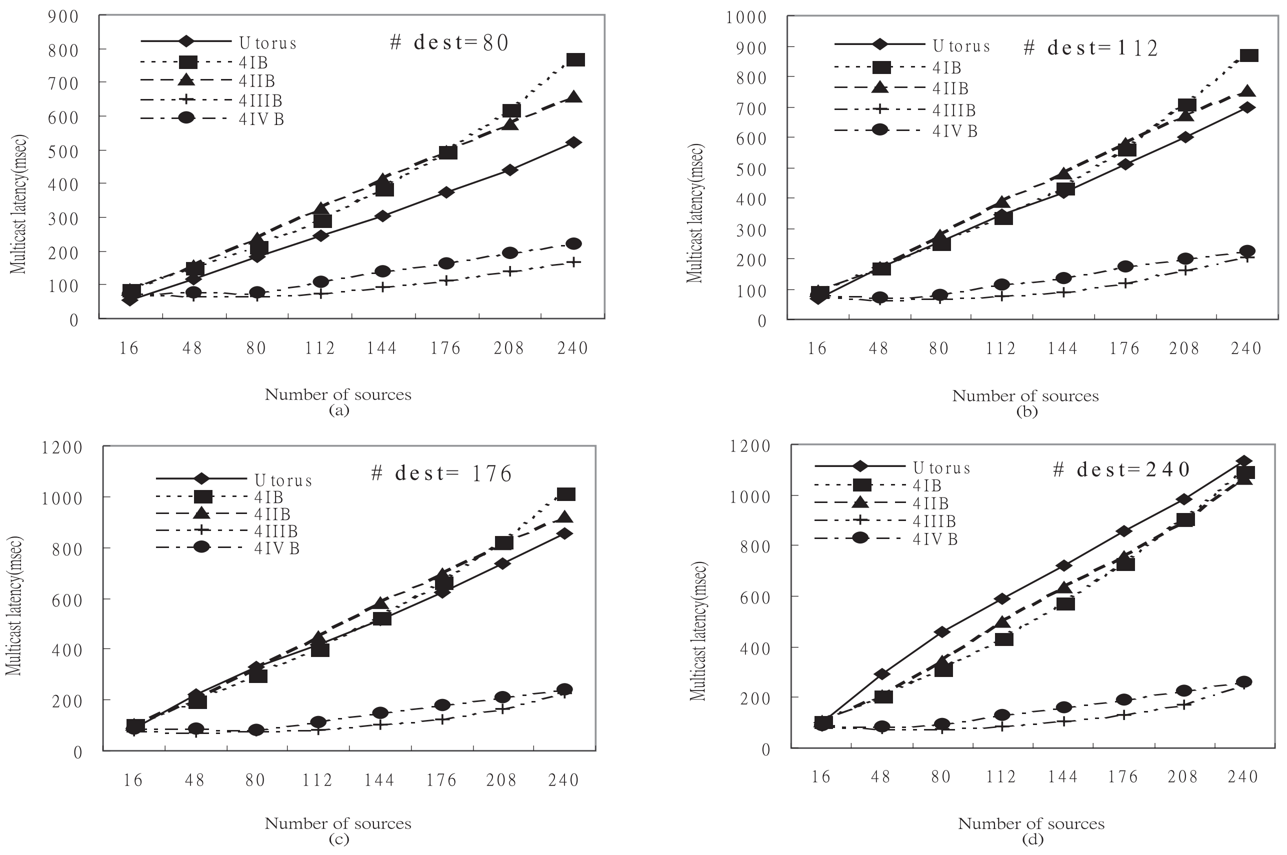


FIGURE 5. Multicast latency in a 16×16 torus at various numbers of sources when there are (a) 80, (b) 112, (c) 176 and (d) 240 destinations ($T_s = 300$ μs, $T_c = 1$ μs and $|M_i| = 32$).

3. Now, multicast is performed in a recursive-doubling manner. Node r_i first sends a multicast message request to x_b which is located at the half of Seq. Then r_i and x_b will be responsible of multicasting M_i to nodes in the first half and the second half, respectively, of Seq recursively.

It is proved in [21] that such an ordering will incur no contention if there is only one multicast in the torus. For instance, consider a multicast $(p_{4,2}, D'_i, M_i)$ in a 8×8 torus, where $D'_i = \{p_{0,3}, p_{1,1}, p_{2,6}, p_{3,4}, p_{5,7}, p_{6,0}, p_{6,4}\}$. After Step 1, we will obtain an ordered list $p_{0,3}, p_{1,1}, p_{2,6}, p_{3,4}, p_{4,2}, p_{5,7}, p_{6,0}, p_{6,4}$. After the left rotation of Step 2, the list becomes $p_{4,2}, p_{5,7}, p_{6,0}, p_{6,4}, p_{0,3}, p_{1,1}, p_{2,6}, p_{3,4}$. Finally, a recursive doubling message delivery as shown in Figure 4a will be performed.

According to [21], when there is only one multicast in the torus, the U-torus scheme takes time

$$T_{ut} = \lceil \log_2 (1 + |D'_i|) \rceil (T_s + LT_c),$$

where T_s is the startup time, T_c is the transmission time per flit and L is the length of M_i. As mentioned earlier, since D'_i is expected to be smaller than D_i, some level of saving can be obtained in addition to the load balancing effect.

4.3. Phase 3: multicasting in DCNs

In this phase, each multicast (r_i, M_i, D'_i) will incur a multicast $(d, M_i, D_i \cap DCN_c)$ on each DCN_c, $c = 0 \ldots \beta - 1$. Since DCN_c is a mesh and dimension-ordered routing is required, one possibility is to apply the *U-mesh* scheme [20]. A review of this scheme is in Section 5.

4.4. Simulation and performance comparison

Since formal analysis is difficult to obtain, we have developed a simulator to study the performance issue. We mainly compared our scheme against the U-torus scheme [21] under various situations. Developed by CSIM18 [29], our simulator monitored communications at the flit level. The parameters used in our simulations are listed below.

- The torus size is 16×16 (machines larger than this were not able to be done due to the capacity of our platform).
- Startup time $T_s = 30$ or 300 μs; transmission time per flit $T_c = 1$ μs. (Note that what is important here is the ratio T_s/T_c, but not their absolute values. For example, in T3D, T_s and T_c are about 1.5 and 0.033 ms respectively, which gives a ratio of 45.)
- Dilation $h = 2$ or 4 (refer to Table 1).
- The problem instance is $\{(s_i, M_i, D_i),\ i = 1 \ldots m\}$ with $|M_i| = 32 \sim 1024$ flits, and $m = |D_i| = 16 \sim 240$ nodes.

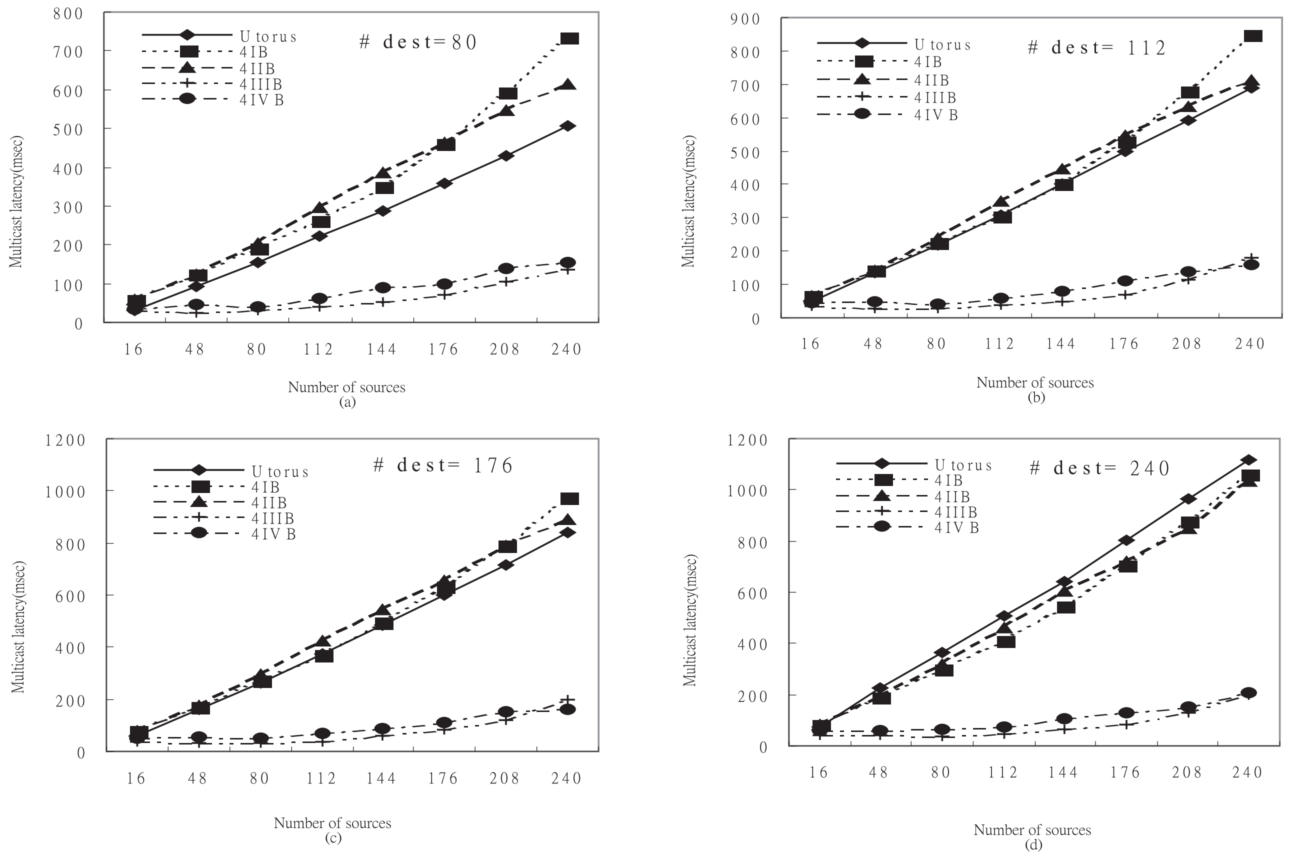


FIGURE 6. Multicast latency in a 16×16 torus at various numbers of sources when there are (a) 80, (b) 112, (c) 176 and (d) 240 destinations ($T_s = 30$ μs, $T_c = 1$ μs and $|M_i| = 32$).

- A hot-spot factor of $p = 25\%, 50\%, 80\%$ or 100% is used. Specifically, when generating D_i, we first choose $p|D_i|$ destination nodes which are common to all destination sets D_i, $i = 1 \dots m$. Then the remaining $(1 - p)|D_i|$ destination nodes are chosen randomly from the network. A larger p thus indicates higher contention on destination nodes.

Below, we show our simulation results from several prospects. We mainly observe the average multicast latency. Based on the subnetworks that are used, our schemes will be denoted as 'HT[B]', where H reflects the value of h, T indicates the type of subnetworks (= I, II, III or IV), and an optional B indicates whether we attempt to achieve load balance in Phase 1 or not. With a B, attempts will be made to evenly distribute multicasts to each DDN and each node in a DDN. If the network type is II or IV, a no-load-balance option is possible by skipping Phase 1 (refer to the discussion in Section 4.1).

Effects of numbers of sources and destinations

Figure 5a shows the multicast latency when $T_s = 300$ μs, $T_c = 1$ μs, $|M_i| = 32$ flits, and $|D_i| = 80$ at various numbers of sources. Undirected subnetworks (Types I and II) have higher latency than that of the U-torus scheme, while directed subnetworks (Types III and IV) have lower latency than that of the U-torus scheme. This is because the latter will utilize more subnetworks, thus giving higher communication parallelism. Generally speaking, subnetworks without link contentions perform better than those with link contentions, so Type I is better than Type II and Type III is better than Type IV. Overall, Type III performs the best.

In Figures 5b, c and d, we enlarge the number of destination nodes to observe the effect. The relative trend remains the same, but the advantage of using our schemes over the U-torus becomes more evident as there are more destinations. When there are 240 destinations (Figure 5d), all our schemes deliver better performance than the U-torus scheme. This shows the importance of load balance especially at high traffic load. When using Type III subnetworks, the performance gain over the U-torus scheme ranges between 2 to 6 times.

Effects of T_s/T_c ratio

We repeated the same simulations as above using a smaller T_s/T_c ratio of 30. Figure 6 shows the results. As compared to Figure 5, we see that the advantage of our schemes over the U-torus scheme becomes slightly larger. Recall that in Phase 1 we have to pay for the costs of re-distributing the multicasts to achieve better load balance. The extra costs in fact reduce as the ratio T_s/T_c decreases.

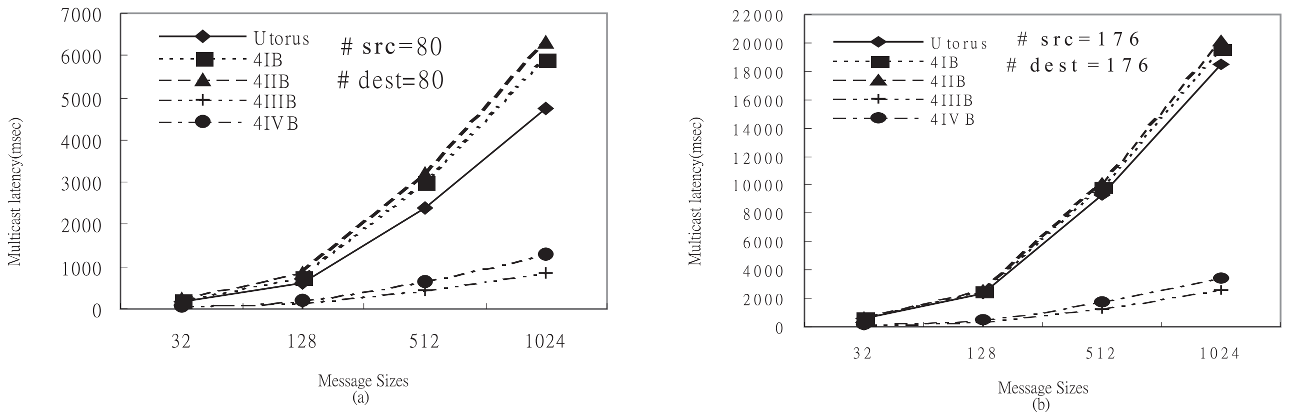


FIGURE 7. Multicast latency in a 16 × 16 torus at various message sizes (a) 80 sources and destinations and (b) 176 sources and destinations ($T_s = 300\ \mu$s and $T_c = 1\ \mu$s).

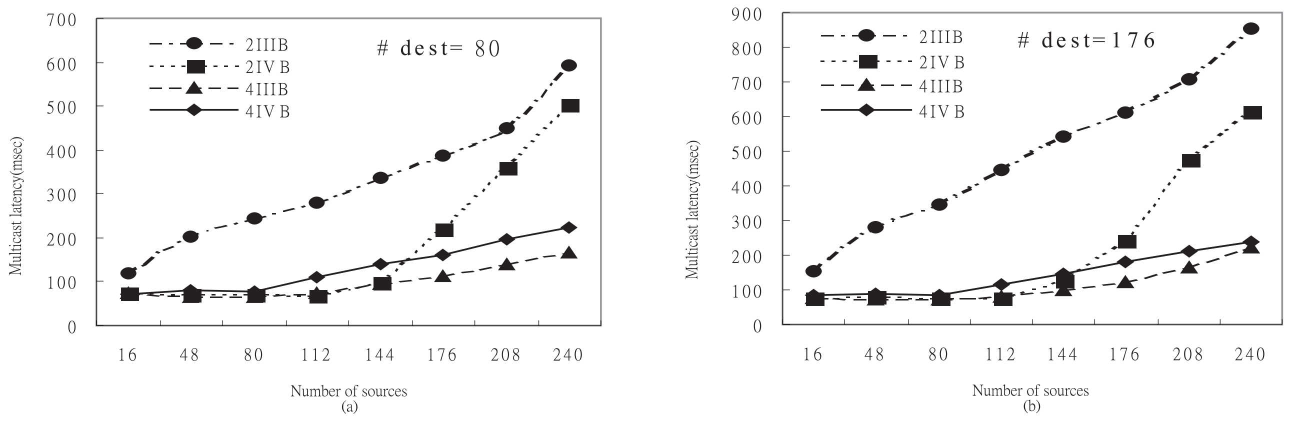


FIGURE 8. Effects of h on multicast latency in a 16 × 16 torus (a) 80 destinations and (b) 176 destinations ($T_s = 300\ \mu$s, $T_c = 1\ \mu$s and $|M_i| = 32$).

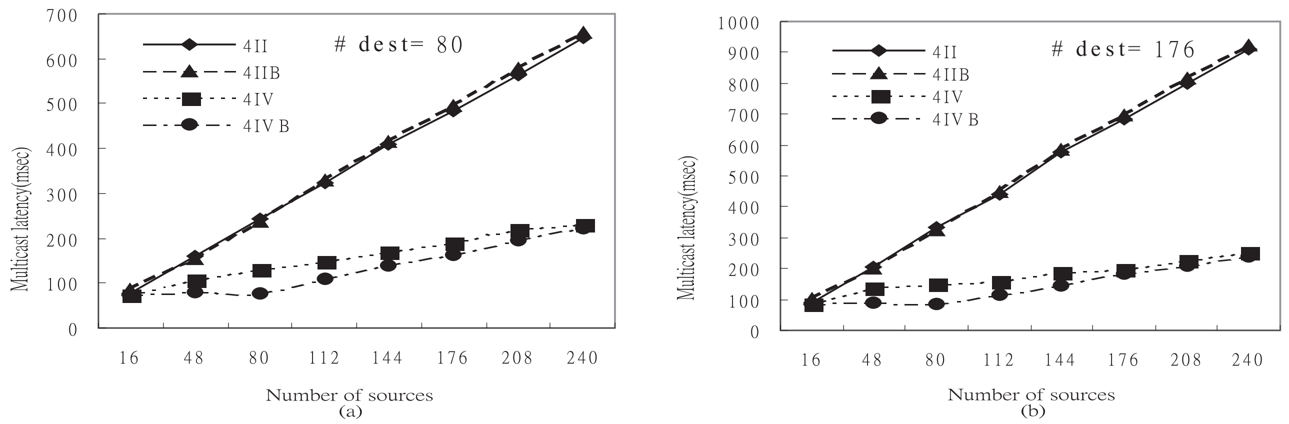


FIGURE 9. Effects of load balance on multicast latency in a 16 × 16 torus (a) 80 destinations and (b) 176 destinations ($T_s = 300\ \mu$s, $T_c = 1\ \mu$s and $|M_i| = 32$).

Effects of message lengths

Figure 7 shows the multicast latency at various message sizes. The gain of our schemes over the U-torus scheme enlarges as message size increases. This again indicates the importance of load balance at heavier traffic load. The same observation applies too if we compare Figure 7a and b (the latter has more sources and destinations). So in subsequent simulations, we will fix the message length at 32 flits, whenever appropriate.

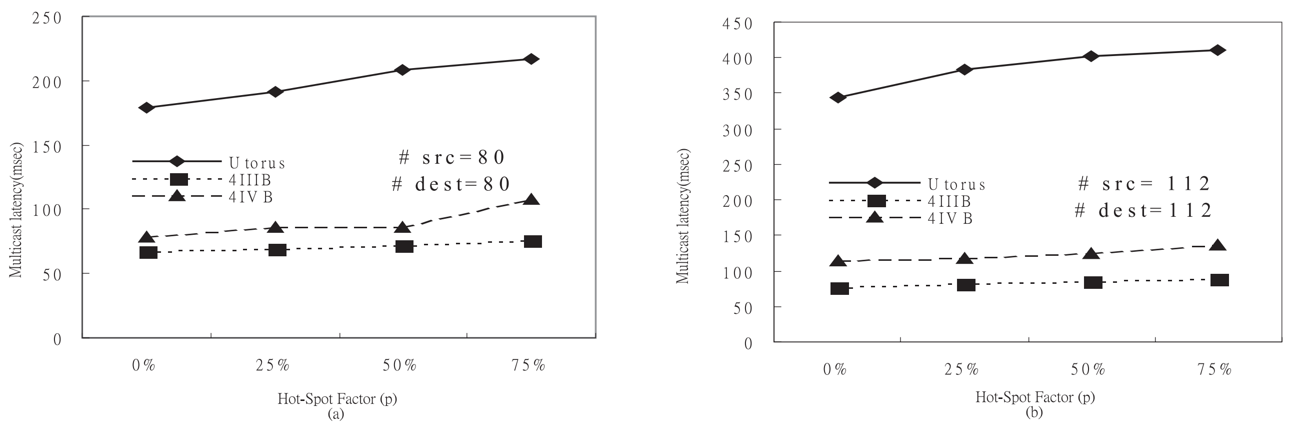


FIGURE 10. Effects of the hot-spot factor on multicast latency in a 16 × 16 torus (a) 80 and (b) 112 sources and destinations ($T_s = 300\ \mu$s, $T_c = 1\ \mu$s and $|M_i| = 32$).

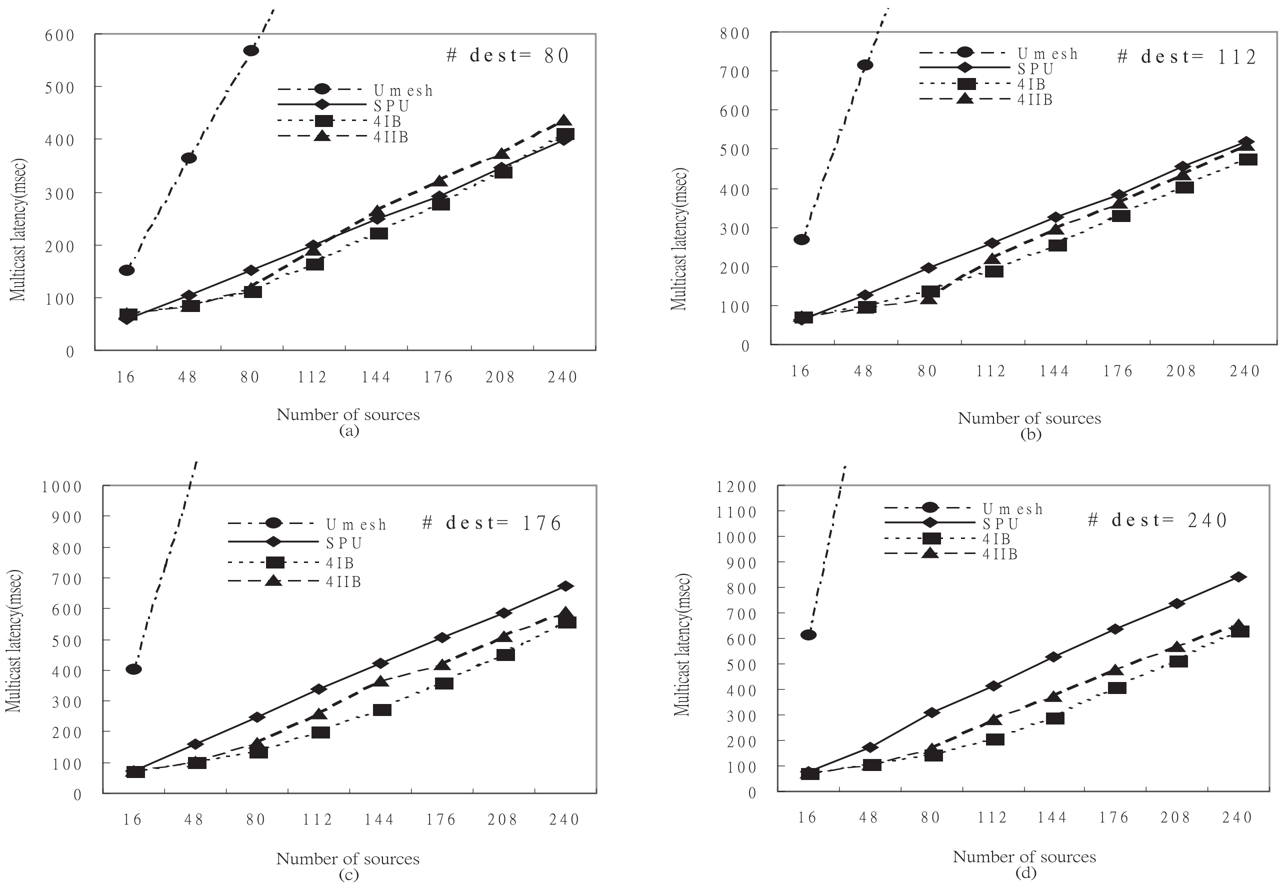


FIGURE 11. Multicast latency in a 16 × 16 mesh at various numbers of sources when there are (a) 80, (b) 112, (c) 176 and (d) 240 destinations ($T_s = 300\ \mu$s, $T_c = 1\ \mu$s and $|M_i| = 32$).

Effects of h

The value of h has two effects. First, it reflects the number of subnetworks, and thus the level of communication parallelism. So a larger h generally delivers better performance. Second, for subnetwork Types II and IV, it reflects the level of link contention, so a smaller h is better for these subnetworks. Figure 8 compares subnetwork Types III and IV when $h = 2$ and 4. The latency trend matches the above observations. One exception is Type 2IVB, which delivers better performance than Type 2IIIB. This is because Type 2IVB offers 4 subnetworks with link contention $h/2 = 1$ (refer to Table 1).

Effects of load balance

As mentioned earlier, subnetwork Types II and IV may be used with a no-load-balance option. Figure 9 shows that the

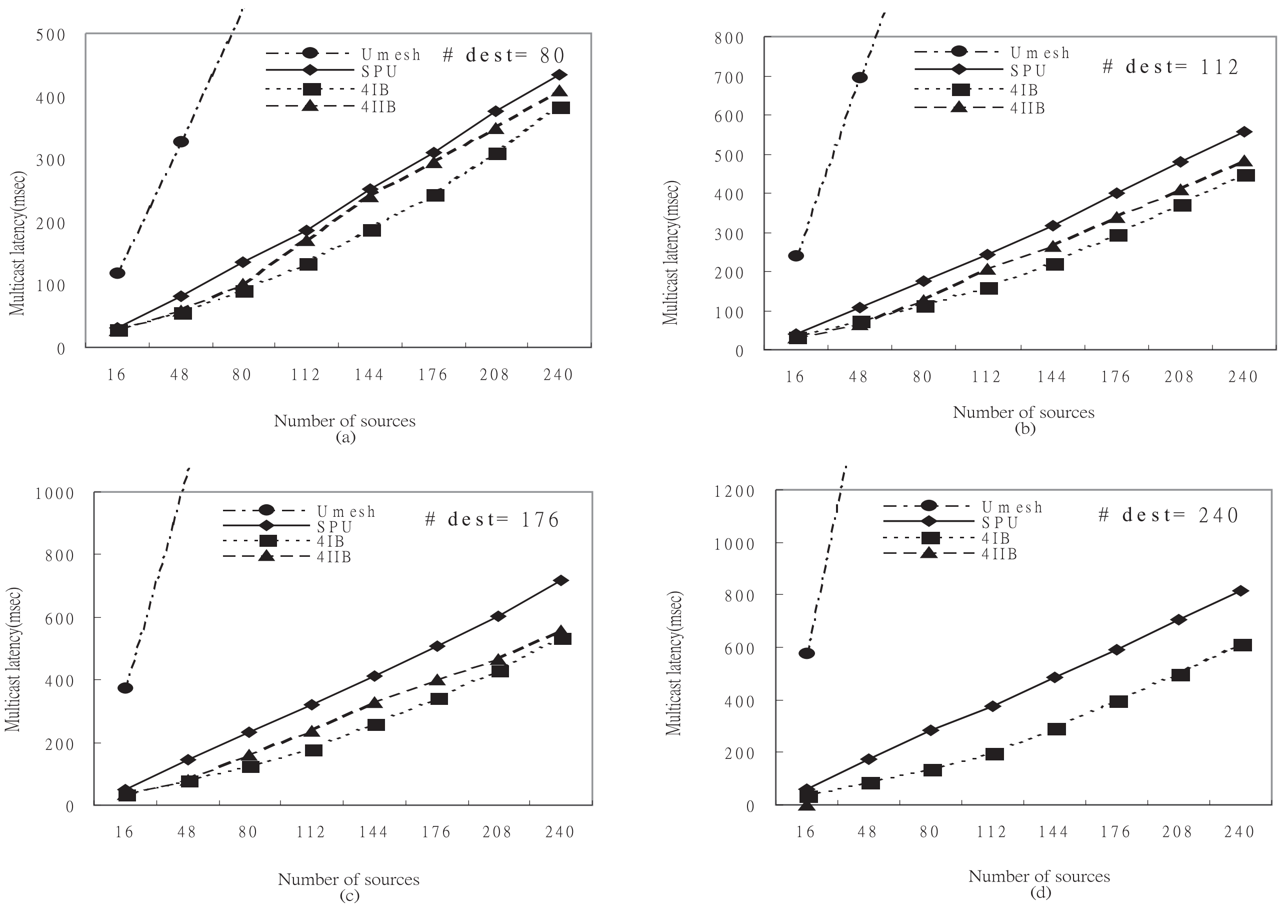


FIGURE 12. Multicast latency in a 16 × 16 mesh at various numbers of sources when there are (a) 80, (b) 112, (c) 176 and (d) 240 destinations ($T_s = 30\ \mu$s, $T_c = 1\ \mu$s and $|M_i| = 32$).

benefit of using load balance is more obvious when there are less sources. With more sources, the benefit is less obvious. In particular, for Type II subnetworks, a no-load-balance option can even deliver slightly better performance when there are $\geq$112 sources. This is because when there are many sources spreading around the network, load balance can be achieved automatically.

Effects of hot-spot factors

Figure 10 shows how the hot-spot factor p affects multicast latency. A larger p will increase the latency. Among the three schemes that are compared, subnetwork Type 4IIIB seems to be most insensitive to the hot-spot effect.

5. MULTI-NODE MULTICAST IN A 2-D MESH

As mentioned earlier, in the case of meshes, we only use undirected subnetworks (Types I and II). The discussion in the previous section for tori can be directly applied, except Phase 2, since DDNs are now (dilated) meshes instead of tori.

5.1. Phase 2: modified multicasting in DDNs

To perform a multicast (r_i, M_i, D'_i) on a mesh, one possibility is to use the U-mesh scheme. There are actually two variations of this scheme [20, 22], which are reviewed below.

The original U-mesh scheme proposed in [20] has only two steps. The first step is the same as that in the U-torus scheme, which sorts the node set $\{r_i, D'_i\}$ in an ascending order by node indices. The second step then performs multicast directly based on this list by a recursive-doubling approach. Taking Figure 4a as an example, the multicast will work as shown in Figure 4b.

As observed in [22], although the U-mesh scheme is congestion free when supporting a single multicast, it will incur intensive node contention when there are multiple multicasts. For multi-node multicasting, it is then suggested to 'left-rotate' each sorted multicast list such that the source node is at the leading position. Interestingly, the adjusted scheme, called source-partitioned U-mesh (SPU), works exactly the same as the U-torus scheme [21], except that now the platform is a mesh. It is shown in [22] that each adjusted multicast itself still remains congestion free and that the contention among multicasts can be reduced significantly.

5.2. Simulation and performance comparison

A simulator was developed to test the performance of our scheme. The same parameters used in torus were used here.

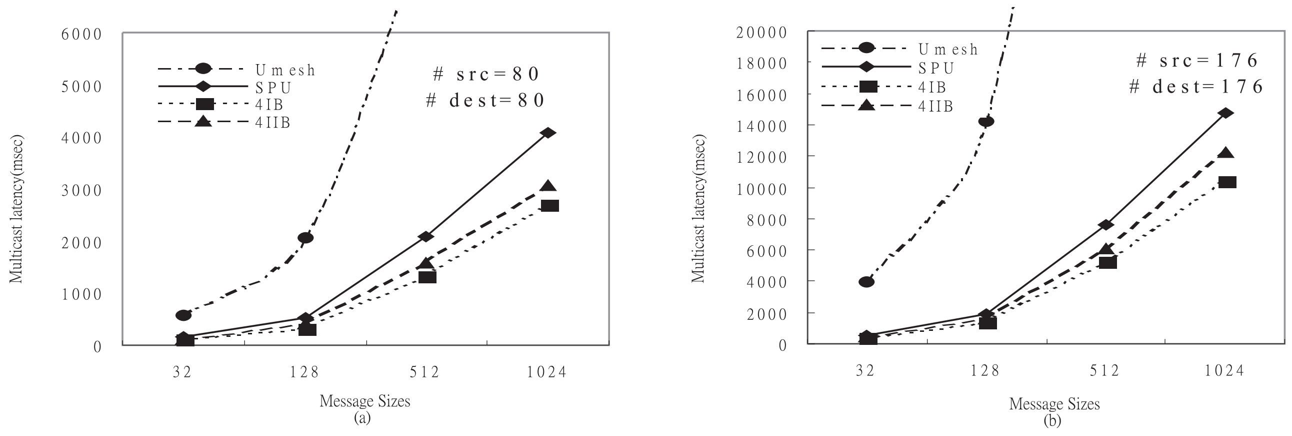


FIGURE 13. Multicast latency in a 16 × 16 mesh at various message sizes (a) 80 sources and destinations, and (b) 176 sources and destinations ($T_s = 300\ \mu$s and $T_c = 1\ \mu$s).

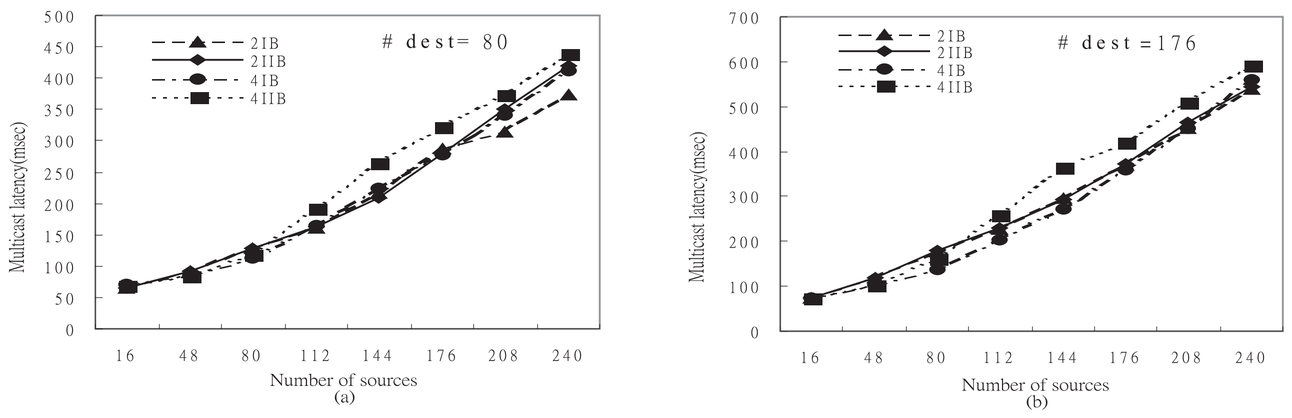


FIGURE 14. Effects of h on multicast latency in a 16 × 16 mesh (a) 80 destinations and (b) 176 destinations ($T_s = 300\ \mu$s, $T_c = 1\ \mu$s and $|M_i| = 32$).

Comparisons were made among the U-mesh, SPU and our schemes.

Effects of numbers of sources and destinations

When there are 80 destinations, Figure 11a shows that both SPU and our schemes outperform the U-mesh scheme with significant gain. Our scheme 4IB starts to outperform the SPU when there are $\geq$80 sources. When there are more destinations, Figures 11b–d reveal the similar trend. But the benefit of our schemes (both Types 4IB and 4IIB) over the SPU becomes more evident. This indicates that our network-partitioning approach is more capable of avoiding contention when load becomes heavier. The improvement on latency over the SPU ranges from 10% to 90%.

Effects of T_s/T_c ratio

Figure 12 repeats the same simulations as above by changing the startup cost $T_s = 30\ \mu$s. Since our schemes need to pay for the costs of redistributing multicasts (Phase 1), a smaller ratio of T_s/T_c will favor our scheme. So Figure 12 shows larger improvement over the SPU as compared to Figure 11.

Effects of message lengths

Similar to the torus case, Figure 13 shows that our schemes are more favorable as message size becomes larger because of the benefit of load balance.

Effects of h

As mentioned earlier, the value of h reflects (i) the level of communication parallelism, and (ii) the level of link contention for Type II. Figure 14 shows that the latter factor plays a more important role. Overall, Type 4IB is the best choice in most cases.

Effects of load balance

Type II may be used with a no-load-balance option. Figure 15 compares Type 4II with 4IIB. The result shows a similar trend as that in a torus—when the number of sources is large, a no-load-balance option may be more favorable.

Effects of hot spots

Figure 16 shows how the hot-spot factor affects multicast latency. All the three schemes tested will have slightly higher multicast latency as the factor increases.

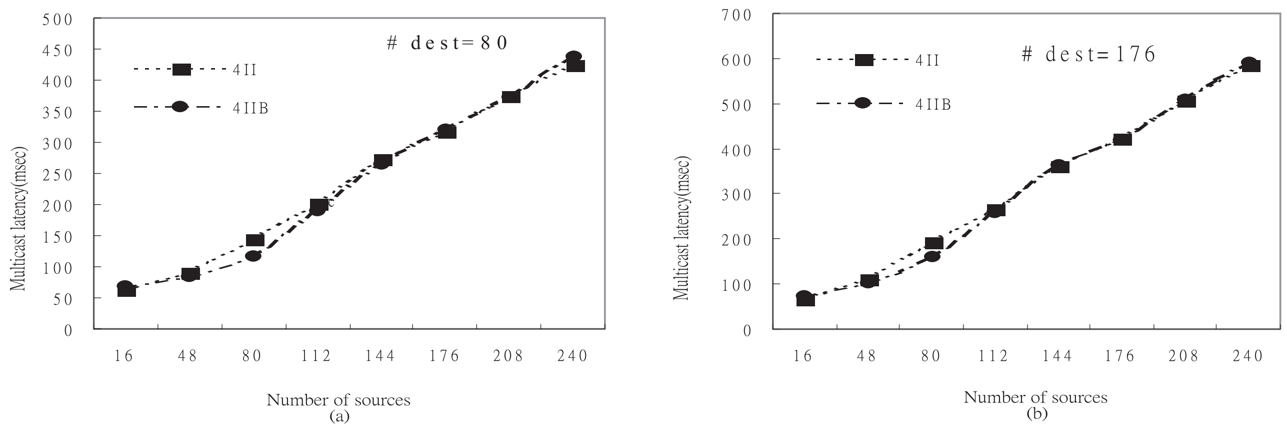


FIGURE 15. Effects of load balance on multicast latency in a 16 × 16 mesh (a) 80 destinations and (b) 176 destinations ($T_s = 300\ \mu$s, $T_c = 1\ \mu$s and $|M_i| = 32$).

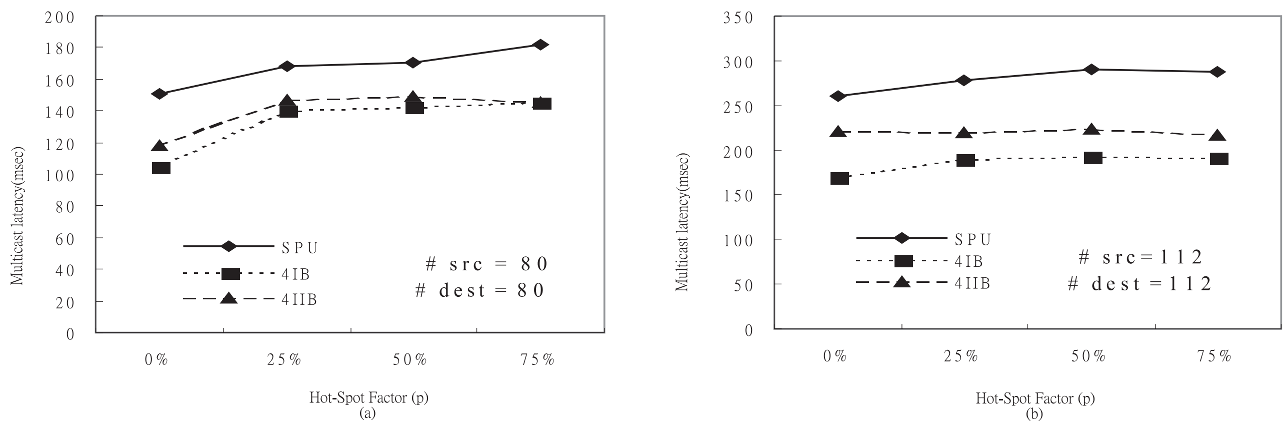


FIGURE 16. Effects of the hot-spot factor on multicast latency in a 16 × 16 mesh (a) 80 and (b) 112 sources and destinations ($T_s = 300\ \mu$s, $T_c = 1\ \mu$s and $|M_i| = 32$).

6. CONCLUSIONS

In this paper, we have developed a set of efficient schemes for multi-node multicast in a torus/mesh. One interesting feature of our approach is that the network is partitioned into several 'dilated' subnetworks to achieve load balance and to increase communication parallelism. Contentions on links and nodes are thus separated evenly to the whole network. Extensive simulations have been conducted, which show significant improvement over existing U-torus, U-mesh and SPU schemes. Our network-partitioning scheme can achieve better load balance and reduce multicast latency, especially when the traffic is heavy or when there exists hot-spot behavior. Since one of the phases (Phase 2) in our schemes is developed based on applying other existing schemes, the possibility still exists of further improving the results in this paper if in the future some better schemes are developed for multi-node multicast in tori and meshes.

ACKNOWLEDGEMENTS

This work is supported by the National Science Council of the Republic of China under Grant NSC89-2213-E-214-023. A preliminary version of this paper has appeared in the *Proceedings of the 2000 International Parallel and Distributed Processing Symposium*.

REFERENCES

[1] Cheng, H. D., Tang, Y. Y. and Suen, C. Y. (1990) Parallel image transformation and its VLSI implementation. *Pattern Recognition*, **23**, 113–129.

[2] Choi, J., Dongarra, J. J., Pozo, R. and Walker, D. W. (1992) ScaLAPACK: a scalable linear algebra library for distributed memory concurrent computers. In *Proc. Symp. on Frontiers of Massively Parallel Computation*, pp. 120–127. IEEE Computer Society Press.

[3] Li, K. and Chaefer, R. (1989) A hypercube shared virtual memory. In *Proc. Int. Conf. on Parallel Processing*, vol. 1, pp. 125–132.

[4] Xu, H., McKinley, P. K. and Ni, L. M. (1992) Efficient implementation of barrier synchronization in wormhole-routed hypercubes multicomputers. *J. Par. Distrib. Comp.*, **16**, 172–184.

[5] Message Passing Interface Forum (1993) *Document for Standard Message-passing Interface*. Technical Report CS-93-214, Department of Computer Science, University of Tennessee, USA.

[6] Bala, V., Bruck, J., Cypher, R., Elustondo, P., Ho, A., Ho, C. T., Kipmis, S. and Snir, M. (1994) CCL: a portable and tunable collective communication library for scalable parallel computers. In *Proc. Int. Parallel Processing Symp.*, Cancun, Mexico, April 1994, pp. 835–843.

[7] Dally, W. J. and Seitz, C. L. (1986) The torus routing chip. *J. Distrib. Comp.*, **1**, 187–196.

[8] Ni, L. M. and McKinley, P. K. (1993) A survey of wormhole routing techniques in directed networks. *IEEE Comp.*, **26**, 62–76.

[9] Intel Corporation (1990) A Touchstone DELTA system description. Intel Corporation. Intel Supercomputing Systems Division, http://nhse.npac.syr.edu/hpccsurvey/orgs/intel/intel.html#Touchstone-Delta.

[10] Foschia, R., Rauber, T. and Runger, G. (1997) Modeling the communication behavior of the Intel Paragon. In *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, pp. 117–124. IEEE Computer Society Press.

[11] Almasi, G. S. and Gottlieb, A. (1994) *Highly Parallel Computing*. Benjamin/Cummings.

[12] Nuth, P. R. and Dally, W. J. (1992) The J-machine network. In *Proc. IEEE Int. Conf. on Computer Design: VLSI in Computer and Processors*, pp. 420–423. IEEE Computer Society Press.

[13] Athas, W. C. and Seitz, C. L. (1988) Multicomputers: message passing concurrent computers. *IEEE Comp.*, **21**(8), 9–24.

[14] Duzett, B. and Buck, R. (1992) An overview of the nCUBE 3 supercomputer. In *Proc. Symp. on Frontiers of Massively Parallel Computation*, pp. 458–464. IEEE Computer Society Press.

[15] Lessler, R. E. and Schwazmeier, J. L. (1993) CRAY T3D: a new dimension for Cray Reasearch. In *COMPCON*, pp. 176–182. IEEE Computer Society Press.

[16] Cray Research Inc. (1995) CRAY T3E scalable parallel processing system. Cray Research Inc. http://www.cray.com/products/systems/crayt3e/.

[17] Boden, N., Cohen, D., Felderman, R., Kulawik, A., Seitz, C., Seizovic, J. and Su, W. (1995) Myrinet—a gigabit per second local area network. *IEEE Macro*, **15**(1), 29–36.

[18] Coster, L. D., Dewulf, N. and Ho, C.-T. (1995) Efficient multi-packet multicast algorithms on meshes with wormhole and dimension-ordered routing. In *Proc. Int. Conf. on Parallel Processing*, vol. 3, pp. 137–141.

[19] Ho, C. T. and Raghunath, M. T. (1992) Efficient communication primitives on hypercubes. *Concurrency: Practice and Experience*, **4**, 427–457.

[20] Mckinley, P. K., Xu, H., Esfahanian, A.-H. and Ni, L. M. (1994) Unicast-based multicast communication in wormhole-routed networks. *IEEE Trans. Par. Distrib. Syst.*, **5**, 1252–1265.

[21] Robinson, D. F., Mckinley, P. K. and Cheng, B. H. C. (1995) Optimal multicast communication in wormhole-routed torus networks. *IEEE Trans. Par. Distrib. Syst.*, **6**, 1029–1042.

[22] Kesavan, R. and Panda, D. K. (1999) Multiple multicast with minimized node contention on wormhole k-ary n-cube networks. *IEEE Trans. Par. Distrib. Syst.*, **10**, 371–393.

[23] Agrawal, N. and Ravikumar, C. P. (1997) An Euler-path-based technique for deadlock-free multicasting. In *Proc. Int. Conf. on Parallel Processing*, pp. 378–383.

[24] Lin, X., McKinley, P. K. and Ni, L. M. (1994) Deadlock-free multicast wormhole routing in 2D mesh multicomputers. *IEEE Trans. Par. Distrib. Syst.*, **5**, 793–804.

[25] Tseng, Y.-C., Yang, M.-H. and Juang, T.-Y. (1998) An Euler-path-based multicasting model for wormhole-routed networks with multi-destination capability. In *Proc. Int. Conf. on Parallel Processing*, pp. 366–373.

[26] Tseng, Y.-C., Wang, S.-Y. and Ho, C.-W. (1999) Efficient broadcasting in wormhole-routed multicomputers: a network-partitioning approach. *IEEE Trans. Par. Distrib. Syst.*, **10**, 44–61.

[27] Wang, S.-Y., Tseng, Y.-C. and Ho, C.-W. (1996) Efficient multicast in wormhole-routed 2d mesh/torus multicomputers: a network-partitioning approach. In *Proc. Symp. on Frontiers of Massively Parallel Computation*, pp. 42–49. IEEE Computer Society Press.

[28] Stamoulis, G. D. and Tsitsiklis, J. N. (1993) Efficient routing schemes for multiple broadcasts in hypercubes. *IEEE Trans. Par. Distrib. Syst.*, **4**, 725–739.

[29] Schwetman, H. D. (1988) Using csim to model complex systems. In *Proc. 1988 Winter Simulation Conf.*, pp. 246–253.