Multi-Node Multicast in Multi-Dimensional Wormhole Tori and Meshes With Load Balance

MING-HOUR YANG*, YU-CHEE TSENG* AND MING-SHIAN JIAN
Department of Computer Science and Information Engineering
National Central University
Chungli, 320 Taiwan

Department of Computer Science and Information Engineering
National Chiao Tung University
Hsinchu, 300 Taiwan
E-mail: yctseng@csie.nctu.edu.tw

This paper considers the multi-node multicast problem in a multi-dimensional wormhole-routed torus/mesh, where there is an arbitrary number of source nodes, each intending to multicast a message to an arbitrary set of destinations. This problem requires a large amount of bandwidth and, thus, typically incurs heavy contention and congestion. Evenly balancing the traffic load around the network is critical to achieving good performance. We show how to use a network-partitioning approach to achieve this goal. Simulation results show significant improvement over existing results in 3D tori and meshes. This work is an extension of our earlier work [22] from 2D tori/meshes to higher dimensional ones.

Keywords: collective communication, interconnection network, load balance, mesh, multicast, parallel processing, torus, wormhole routing

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 one destination. Applications of collective communication include parallel numerical algorithms, graph algorithms, image processing algorithms, cache coherence, and barrier synchronization [5, 6, 13, 23].

Wormhole routing [8, 16] is becoming a natural switching technology for interconnection networks. It is characterized by low communication latency and is quite insensitive to the routing distance in the absence of link contention. This technology has been adopted by the Intel Touchstone DELTA [1], Intel Paragon, MIT J-machine [17], Caltech MOSAIC [4], nCUBE 3 [9], Cray T3D and T3E [2, 12], and Myrinet.

This paper considers the multi-node multicast problem in a multi-dimensional torus/mesh with wormhole, dimension-ordered, and one-port routing capabilities. There is
an arbitrary number of source nodes, each intending to send a multicast message to an arbitrary set of destination nodes. The challenge is that there is no global knowledge of these multicast patterns, and there may exist serious contention when the source set or destination set is large or when the hot-spot effect exists (i.e., sources and/or destinations concentrate in a small 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 re-routing them, to these subnetworks, with the expectation of balancing the traffic load around the network links.

In the literature, much attention has been paid to the single-node multicast problem [7, 10, 15, 18]. The authors in [10] suggested connecting destination nodes to form a Hamiltonian path, on which the multicast message can be sent in a pipelined manner. A widely used approach is the U-torus (resp., U-mesh) scheme proposed in [15] (resp., [18]), in which destination nodes are connected as a binomial tree. With only one multicast, routing is guaranteed to be congestion-free. The idea was further generalized in [7], where a Fibonacci tree was used instead. Works on multi-node multicast include [11] (for meshes with dimension-ordered routing) and [3, 14, 21] (for tori/meshes with non-dimension-ordered routing). The authors in [11] showed how to modify the U-mesh scheme to relieve the contention problem when there are multiple multicasts.

This work is an extension of our previous work [22], which focused on multi-node multicast on a 2D torus/mesh. Here, we extend it to higher dimensional tori/meshes. We propose to partition a torus/mesh into a number of subnetworks, each maintaining a similar torus/mesh structure after partitioning. Now given a multi-node multicast pattern, we first “delegate” each multicast to a subnetwork, with the hope of balancing the traffic load on each subnetwork. Then, on each subnetwork, about an equal number of multicasts are performed. Based on our prior experience [22], a significant gain can be obtained in this way. Thus, it is worth investigating this possibility on 3D and higher dimensional torus/mesh networks. In this paper, several ways to partition a torus/mesh are proposed. Based on extensive simulations, we show that our network-partitioning approach can achieve a better load balance and in many cases improve over the U-mesh/U-torus schemes [11, 15, 18]. One interesting case that we will explore is the hot-spot behavior, where multicast sources and destinations may concentrate in a small area of the network. According to our observations, load balancing plays a more important role in such cases.

The rest of this paper is organized as follows. Preliminaries are given in section 2. Ways to partition a 3D torus/mesh are presented in section 3. Routing algorithms and simulation results on 3D tori and meshes are given in section 4 and section 5, respectively. Section 6 extends our result to higher dimensional tori/meshes. Conclusions are drawn in section 7.

2. PRELIMINARIES

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 [16]. 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 along all its ports simultaneously. The generic architecture of a wormhole router and a torus connected by routers is shown in Fig. 1.

![Fig. 1. A 4 × 4 torus and the generic node architecture for wormhole routing.](image)

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 $T_s + LT_c$ [16], where $T_s$ is the startup time (for initializing the communication) and $T_c$ is the transmission time per byte.

In this paper, we consider networks that are connected as torus or mesh of any dimension.

### 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, Fig. 2 shows two subnetworks, $G_0$ and $G_1$, in an $8 \times 8 \times 8$ 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 $(x, y, z) \in C'$. Then the vertices $x$, $y$ and $z$ are not necessarily in the vertex set $V'$. For instance, in Fig. 2, the subnetwork $G_0$ contains links $(p_{0,0,0}, p_{0,1,0})$ and $(p_{0,1,0}, p_{0,2,0})$. However, node $p_{0,1,0}$ is not in $G_0$’s node set.
- The previous point, in fact, carries special significance for wormhole routing. For instance, each $G_i$ in Fig. 2 can be considered as a $4 \times 4 \times 4$ torus, with each link “dilated” by two 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 the links in its link set, should be constrained in its capability in initiating/retrieving packets into/from the subnetwork.
Fig. 2. Two dilated-2 subnetworks, each as an undirected $4 \times 4 \times 4$ torus, in an $8 \times 8 \times 8$ torus.

subject to its node set. For instance, in Fig. 2, nodes $p_{0,1,0}$ and $p_{0,3,0}$ of $G_0$ are neither allowed to initiate a new worm into, nor allowed to retrieve a pass-by worm from, the subnetwork. They can only passively relay worms it receives 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 the important issue of making each subnetwork less dependent on other subnetworks, as formulated in the following definition.

Definition 2 Given two subnetworks $G_1 = (V_1, C_1)$ and $G_2 = (V_2, C_2)$, $G_1$ and $G_2$ are said to be node-contention-free if $V_1 \cap V_2 = \emptyset$, and link-contention-free if $C_1 \cap C_2 = \emptyset$.

Intuitively, freedom from 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 from 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 $G_1$, $G_2$, ..., $G_k$, the level of node contention (resp., level of link contention) among these subnetworks is defined as the maximum number of times that a node (resp., link) appears in these subnetworks, among all the nodes (resp., links) in the network.
2.3 General Model for Multi-Node Multicasts

A multi-node multicast instance can be denoted by a set of 3-tuple \( \{(s_i, M_i, D_i) \}_{i=1..m} \). There are \( m \) source nodes \( s_1, s_2, \ldots, s_m \). Each \( s_i, i = 1..m \), intends to multicast a message \( M_i \) to a set \( D_i \) of destinations. However, the multicast pattern is not globally known to any node. 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 from \( G \) two kinds of subnetworks: data-distributing networks (DDNs) and data-collecting networks (DCNs). Suppose we have \( \alpha \) DDNs, \( DDN_0, DDN_1, \ldots, DDN_{\alpha-1} \), and \( \beta \) 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 together contain all the nodes of \( G \).
- **P3**: The vertex sets of \( DDN_i \) and \( DCN_j \) intersect at least one node, for all \( 0 \leq i < \alpha \) and \( 0 \leq j < \beta \).

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

**Phase 1**: Each multicast \((s_i, M_i, D_i), i = 1..m\), selects a target data distribution network, say, \( DDN_a \) to distribute its message. This selection should be made 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 \( r_i \), multicast \((r_i, M_i, D'_i)\) is performed on \( DDN_a \), where the destination set \( D'_i \) is obtained from \( D_i \) by means of the following transformation. For each \( DCN_b, b = 0..\beta - 1, \) if \( DCN_b \) contains one or more destination nodes in \( D_i \), then 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 add \( d \) into \( D'_i \).

**Phase 3**: In each \( DCN_b \), \( b = 0..\beta - 1, \) 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 for distributing multicast messages around the network, while DCNs 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. Therefore, 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 necessary, 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 DDNs and DCNs in tori and meshes that satisfy our needs.
3. SUBNETWORKS OF A 3D TORUS/MESH

We will first discuss how to find DDNs and DCNs in a torus. Then, we will briefly summarize how the definitions can be modified for a mesh.

3.1 DDN’s and DCN’s in a 3D Torus

A 3D torus $T_{s,t,r}$ consists of $s \times t \times r$ nodes, each denoted as $p_{x,y,z}$, where $0 \leq x < s$, $0 \leq y < t$, and $0 \leq z < r$. Node $p_{x,y,z}$ has links connecting it to six other nodes: $p_{(x \pm 1) \bmod s,y,z}$, $p_{x,(y \pm 1) \bmod t,z}$, and $p_{x,y,(z \pm 1) \bmod r}$. In Definitions 4-9 below, we define six types of DDN’s in a 3D torus.

Definition 4 Given a torus $T_{s,t,r}$ and any integer $h$ that divides $s$, $t$, and $r$, define $h$ subnetworks $G_i = (V_i, C_i)$, $i = 0 \ldots, h-1$, such that

$$V_i = \{p_{x,y,z} | x = ah + i, y = bh + i, z = ch + i,$$
$$\text{for all } a = 0 \ldots \frac{s}{h} - 1, b = 0 \ldots \frac{t}{h} - 1 \text{ and } c = 0 \ldots \frac{r}{h} - 1\},$$

$$C_i = \{\text{all channels forming a line along the x-axis and crossing } p_{0,bh+ih},c\},$$
$$\text{all channels forming a line along the y-axis and crossing } p_{ah+i,0,ch},i+1,\text{ and }$$
$$\text{all channels forming a line along the z-axis and crossing } p_{ah+i,bh+i,0}\}.$$

Intuitively, $G_0$ contains all nodes of the form $p_{bh,ch}$, and $G_i$ is obtained from $G_0$ by shifting $G_0$’s nodes by $i$ positions on all indices. In our terminology, each subnetwork is a “dilated-h” torus of size $(s/h) \times (t/h) \times (r/h)$. Fig. 2 shows an example with two dilated-2 $4 \times 4 \times 4$ tori in an $8 \times 8 \times 8$ torus ($h = 2$).

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

However, the subnetworks defined in Definition 4 do not use all of the links in the original torus (e.g., links $(p_{0,1,0}, p_{1,1,0})$ and $(p_{1,1,0}, p_{2,1,0})$). The following definition tries to utilize these unused links by adding more subnetworks without increasing node contention.

Definition 5 Given a torus $T_{s,t,r}$ and any integer $h$ that divides $s$, $t$, and $r$, define $h^2$ subnetworks $G_{i,j} = (V_{i,j}, C_{i,j})$, $i, j = 0 \ldots, h-1$, such that

$$V_{i,j} = \{p_{x,y,z} | x = ah + i, y = bh + i, z = ch + j,$$
$$\text{for all } a = 0 \ldots \frac{s}{h} - 1, b = 0 \ldots \frac{t}{h} - 1 \text{ and } c = 0 \ldots \frac{r}{h} - 1\},$$

$$C_{i,j} = \{\text{all channels forming a line along x-axis and crossing } p_{0,bh+ih},c\},$$
$$\text{all channels forming a line along y-axis and crossing } p_{ah+i,0,ch+j},\text{ and }$$
$$\text{all channels forming a line along z-axis and crossing } p_{ah+i,bh+i,0}\}.$$

Fig. 3 shows an example with four dilated-2 $4 \times 4 \times 4$ subnetworks in an $8 \times 8 \times 8$ torus ($h = 2$). Intuitively, $G_{i,i}$, $i = 0 \ldots, h-1$, are the subnetworks defined in. For each $G_{i,i}$, we shift its nodes along the third dimension to obtain $h-1$ subnetworks $G_{i,j}$, $i \neq j$. 
Observe that in Definition 5, all the 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 $p_{1,0,0}$ and $p_{0,1,0}$) 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 6** Given a torus $T_{s\times t\times r}$ and any integer $h$ that divides $s$, $t$ and $r$, define $h^3$ subnetworks $G_{i,j,k} = (V_{i,j,k}, C_{i,j,k})$, $i, j, k = 0..h-1$, such that

$$V_{i,j,k} = \{p_{x,y,z} \mid x = ah+i, y = bh+j, z = ch+k,$$
$$\text{for all } a = 0..\frac{s}{n}-1, b = 0..\frac{t}{n}-1 \text{ and } c = 0..\frac{r}{n}-1\},$$

$$C_{i,j} = \{\text{all channels forming a line along x-axis and crossing } p_{0,bh+j,ch+k},$$
$$\text{all channels forming a line along y-axis and crossing } p_{ah+i,0,ch+k},$$
$$\text{and all channels forming a line along z-axis and crossing } p_{ah+i,bh+j,0}\}.$$
Definition 7 Given a torus $T_{s,t,r}$ and any integer $h$ that divides $s$, $t$ and $r$, define $h$ subnetworks $G^+_i = (V^+_i, C^+_i), i = 0..h - 1$, such that (refer to Definition 4)
\[
V^+_i = V_i, \\
C^+_i = \{\text{all positive links in } C_i\},
\]
and $h$ subnetworks $G^-_i = (V^-_i, C^-_i), i = 0..h - 1$, such that
\[
V^-_i = \{ p_{x,y,z} | x = ah + i + \delta, y = bh + i, z = ch + j, \\
\text{for all } a = 0..\frac{s}{h} - 1, b = 0..\frac{t}{h} - 1 \text{ and } c = 0..\frac{r}{h} - 1\}, \\
C^-_i = \{\text{all channels forming a line along x-axis and crossing } p_{0,bh+i,ch+j}, \\
\text{all channels forming a line along y-axis and crossing } p_{ah+i,0,ch+j}, \text{ and} \\
\text{all channels forming a line along z-axis and crossing } p_{ah+i,0,bh+j}\},
\]
where $\delta$ is any constant satisfying $1 \leq \delta \leq h - 1$.

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 first dimension by $\delta$ positions and using only negative links (this is to resolve the node contention). For instance, Fig. 4 illustrates this definition in a $8 \times 8 \times 8$ torus with $h = 2$ and $\delta = 1$ (for clarity, the four 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 (without this property, routing would be difficult).

Lemma 3 The $2h$ subnetworks $G^+_i$ and $G^-_i, i = 0..h - 1$, defined in Definition 7, are free from both node and link contention.

In the following are given extensions of Definition 5 and Definition 6.

Definition 8 Given a torus $T_{s,t,r}$ and any integer $h$ that divides $s$, $t$ and $r$, define $h^2$ subnetworks $G^+_{i,j} = (V^+_{i,j}, C^+_{i,j}), i, j = 0..h - 1$, such that (refer to Definition 5)
\[
V^+_{i,j} = V_{i,j}, \\
C^+_{i,j} = \{\text{all positive links in } C_{i,j}\},
\]
and $h^2$ subnetworks $G^-_{i,j} = (V^-_{i,j}, C^-_{i,j}), i, j = 0..h - 1$, such that
\[
V^-_{i,j} = \{ p_{x,y,z} | x = ah + i + \delta, y = bh + i, z = ch + j, \\
\text{for all } a = 0..\frac{s}{h} - 1, b = 0..\frac{t}{h} - 1 \text{ and } c = 0..\frac{r}{h} - 1\}, \\
C^-_{i,j} = \{\text{all channels forming a line along x-axis and crossing } p_{0,bh+i,ch+j}, \\
\text{all channels forming a line along y-axis and crossing } p_{ah+i,0,ch+j}, \text{ and} \\
\text{all channels forming a line along z-axis and crossing } p_{ah+i,0,bh+j}\},
\]
where $\delta$ is any constant satisfying $1 \leq \delta \leq h - 1$. 

**Definition 9** Given a torus $T_{s\times t\times r}$ and any integer $h$ that divides $s$, $t$ and $r$, define $h^3$ subnetworks $G_{i,j,k}^h = (V_{i,j,k}^h, C_{i,j,k}^h)$, $i, j, k = 0..h-1$, such that (refer to Definition 6)

$$V_{i,j,k} = V_{i,j,k}^h,$$

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

**Lemma 4** The $2h^2$ (resp., $h^3$) subnetworks $G_{i,j}^+$ and $G_{i,j}^-$, $i, j = 0..h-1$ (resp., $G_{i,j,k}^+ - G_{i,j,k}$, $i, j, k = 0..h-1$), defined in Definition 8 (resp., Definition 9), are free from node contention but have link contention of $h$ (resp., $h^2/2$).
In Table 1, we summarize the above definitions based on the levels of node and link contention incurred by different subnetworks. In all cases, there is no node contention. In general, the level of link contention is proportional to the number of subnetworks divided by \( h \). Also, the maximum number of subnetworks that can be obtained without incurring node and link contention is \( 2^h \), as given by Definition 7.

### Table 1. Comparison based on levels of node and link contention incurred by different subnetworks.

<table>
<thead>
<tr>
<th>type</th>
<th>Subnet.</th>
<th>no of subnet.</th>
<th>links</th>
<th>node cont.</th>
<th>Link cont.</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>( G_{i}, i = 0..h - 1 )</td>
<td>( h )</td>
<td>undirected</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>II</td>
<td>( G_{i,j}, i, j = 0..h - 1 )</td>
<td>( h^2 )</td>
<td>undirected</td>
<td>no</td>
<td>( h )</td>
</tr>
<tr>
<td>III</td>
<td>( G_{i,j,k}, i, j, k = 0..h - 1 )</td>
<td>( h^3 )</td>
<td>undirected</td>
<td>no</td>
<td>( h^3 )</td>
</tr>
<tr>
<td>IV</td>
<td>( G_{i}^{+}, G_{i}^{-}, i = 0..h - 1 )</td>
<td>( 2h )</td>
<td>directed</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>V</td>
<td>( G_{i}^{+}, G_{i}^{-}, i, j = 0..h - 1 )</td>
<td>( 2h^2 )</td>
<td>directed</td>
<td>no</td>
<td>( h )</td>
</tr>
<tr>
<td>VI</td>
<td>( G_{i,j,k}, i, j, k = 0..h - 1 )</td>
<td>( h^3 )</td>
<td>directed</td>
<td>no</td>
<td>( h^3/2 )</td>
</tr>
</tbody>
</table>

**Definition 10** Given a torus \( T_{s\times t\times r} \) and any integer \( h \) that divides both \( s \) and \( t \), define \( s\times t\times r/h^3 \) data collecting networks \( DCN_{a,b,c} = (V_{a,b,c}, C_{a,b,c}) \), \( a = 0..s/h - 1 \), \( b = 0..t/h - 1 \) and \( c = 0..r/h - 1 \), such that

\[
V_{a,b,c} = \{ p_{x,y,z} \mid x = a \times h + i, y = b \times h + j, z = c \times h + k \text{ for all } i, j, k = 0..h - 1 \},
\]

\[
C_{a,b,c} = \{ \text{(undirected) links induced by } V_{a,b,c} \}\]

Intuitively, we simply partition the torus into a number of submeshes, each of size \( h \times h \times h \). For instance, when \( h = 2 \), there are 64 DCNs (each as a cube) in an \( 8 \times 8 \times 8 \) torus (the first \( DCN_{0,0,0} \) is shown in Fig. 2). The same DCN definition will be applied to all the above subnetworks. Finally, it is not hard to see that these definitions satisfy properties P1-P5.

### 3.2 DDN’s and DCN’s in a 3D Mesh

A 3D mesh \( M_{s\times t\times r} \) is similar to a torus \( T_{s\times t\times r} \) except that there are no “wrap-around” links. The above six DDN definitions for a torus can be easily translated to the mesh case by excluding 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. Therefore, we will only consider using Definition 4, Definition 5 and Definition 6 to define our DDNs. The levels of node and link contention remain the same. Also, the same DCNs in Definition 10 are used.

### 4. MULTI-NODE MULTICAST IN A 3D TORUS

Given a multi-node multicast instance \( \{ (s_i, M_i, D_i), i = 1..m \} \), we will next show in more detail how the multi-node multicast model described in section 2.1 can be applied using the DDNs and DCNs defined above. Throughout this section, we will let \( DDN_0 \),
MULTI-NODE MULTICAST IN TORI AND MESHES

$DDN_1, \ldots, DDN_{\alpha-1}$ be $\alpha$ DDNs obtained from Definition 4, 5, 6, 7, 8, or 9, and let $DCN_0, DCN_1, \ldots, DCN_{\beta-1}$ be $\beta$ DCNs obtained from Definition 10.

4.1 Phase 1: Balancing Traffic Among DDNs

In this phase, each multicast $(s_i, M_i, D_i)$, $i = 1..m$, should be distributed to one of the DDNs. There are two concerns related to distributing the load. First, each DDN should receive about the same number of multicasts. Second, in each DDN, each node should be responsible for about the same number of multicasts. If the multicast pattern is given in advance, these goals are not hard to achieve.

A more distributed approach is to have each $s_i$ randomly choose a DDN as its target subnetwork. This approach is more appropriate if multicasts arrive in an unpredictable or asynchronous manner, or as in a stochastic model, such as that assumed in [20]. In particular, if subnetwork types III and VI 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 is achieved automatically if multicasts arrive stochastically randomly.

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 noted that the way in which $D_i$ is translated into $D'_i$ will incur a concentration effect, thus, there is a high probability that $|D'_i| < |D_i|$. Therefore, the multicast is on a smaller network with a smaller destination set. Statistically, we can say that $|D'_i| = |D_i| / \alpha$.

Overall, each DDN will still need to perform a multi-node multicast. Improvement is expected if Phase 1 can produce a load-balancing effect. With the dimension-ordered routing constraint, one possibility is to use the U-torus scheme [18] 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 \{ri, Di\}. We first sort the nodes in the set in ascending order as follows. Nodes in a torus are totally ordered according to their indices such that $p_{ij} < p_{i'j'}$ if $i < i'$ or $i = i'$ but $j < j'$.
2. Let the sorted list be $x_0, x_1, \ldots, x_n$, where $n = 1 + |D'|$. Also let $x_0 = r_i$. Then we "left-rotate" the list into $Seq = x_0, x_{n+1}, \ldots, x_n, x_0, x_1, \ldots, x_{n-1}$ (i.e., $r_i$ becomes the leading element).
3. Now, multicast is performed in a recursive-doubling manner. Node $r_i$ first sends a multicast message request to $x_0$, which is located at the half of $Seq$. Then $r_i$ and $x_0$ are responsible for multicasting $M_i$ to nodes in the first half and the second half, respectively, of $Seq$ recursively.

According to [18], when there is only one multicast in the torus, the U-torus scheme takes time
\[ T_{st} = \left\lceil \log_2 (1 + |D'|) \right\rceil (T_s + LT_c) , \]

where \( T_s \) is the startup time, \( T_c \) is the transmission time per flit, and \( L \) is the length of \( M_c \).

As mentioned earlier, since \( D' \) is expected to be smaller than \( D_c \), some amount of savings can be obtained in addition to the load balancing effect.

4.3 Phase 3: Multicasting in DCNs

In this phase, each multicast \( (r_i, M_i, D'_i) \) incurs a multicast \( (d, M_i, D_i \cap DCN_c) \) on each \( DCN_c, c = 0..\beta - 1 \). Since \( DCN_c \) is a mesh and dimension-ordered routing is required, one possibility is to apply the SPU scheme [11]. This scheme is reviewed in section 5.

4.4 Simulation and Performance Comparison

Since formal analysis is difficult to conduct, we have developed a simulator to study the performance issue. We mainly compared our scheme against the U-torus scheme [18] under various situations. Developed by CSIM18 [19], our simulator monitored communications at the flit level. The parameters used in our simulations are listed below:

- torus size \( 8 \times 8 \times 8 \) or \( 16 \times 16 \times 16 \);
- startup time \( T_s = 30 \) or \( 300 \mu \text{sec} \), transmission time per flit \( T_c = 1 \mu \text{sec} \);
- dilation \( h = 2 \) or \( 4 \) (refer to Table 1).
- The problem instance was \( \{(s_i, M_i, D_i), i = 1..m\} \) with \( |M| = 32 \sim 1024 \) flits, and \( m = |D| = 160 \sim 510 \) nodes. All these multicasts were generated simultaneously, except in experiment part F.
- A hot-spot factor of \( p = 5\%, 25\%, 50\%, 75\%, \) or \( 90\% \) was used. A larger \( p \) value indicated higher contention on source or destination nodes. Specifically, we define three types of hot-spots
  - Source hot-spot: We choose \( p |s_i| \) source nodes, which are adjacent. Then, the remaining \( (1 - p)|s_i| \) source nodes are chosen randomly from the network.
  - Destination hot-spot: We choose \( p |D_i| \) destination nodes, which are adjacent and are common to all destination sets \( D_i, i = 1..m \). Then, the remaining \( (1 - p)|D_i| \) destination nodes are chosen randomly.
  - Source-destination hot-spot: This contains both a source hot-spot and a destination hot-spot, with a factor of \( p \).

Below, we give our simulation results from several prospects. We mainly observed the average multicast latency. Based on the subnetworks that were used, our schemes were denoted as “HT,” where \( H \) reflects the value of \( h \) and \( T \) indicates the type of sub-network (= I, II, III, IV, V or VI).

A) Effects of the Numbers of Sources and Destinations: Fig. 5 (a) shows the multicast latency when \( T_s = 300\mu \text{sec}, T_c = 1\mu \text{sec}, |M| = 32 \) flits, and \( |D| = 160 \) for various numbers of sources. Our types I and II had the highest latency, followed by our types IV and V, which in turn followed by our type VI. The U-torus and our type III had the lowest latency. Since the traffic load was not heavy, the
Fig. 5. Multicast latency in an $8 \times 8 \times 8$ torus for various numbers of sources when there were: (a) 160, (b) 256, and (c) 510 destinations ($T_s = 300\mu\text{sec}$, $T_c = 1\mu\text{sec}$, and $|M| = 32$). Note that we have ordered the logos based on the performance of the corresponding schemes.

U-torus performed pretty well because it did not spend time re-distributing the multicasts as ours did. Generally speaking, among our schemes the more sub-networks we used, the lower the latency was.

As shown in Fig. 5 (b), we increased the number of destinations to 256. Since the traffic load was now higher, the latency incurred by the U-torus became relatively higher than that of some of our schemes. This became more apparent as we further increased the number of destinations to 510, as shown in Fig. 5 (c).

B) Effects of the $T_s/T_c$ Ratio: We repeated the same simulations in part A using a smaller $T_s/T_c$ ratio of 30. Fig. 6 shows the results. As compared to Fig. 5, the trend remained unchanged.

C) Effects of Message Lengths: Fig. 7 shows the multicast latency for various message sizes. The gain achieved by our schemes over the U-torus scheme enlarged as the message size increased. This again indicates the importance of load balance at heavier traffic loads.

D) Effects of $h$: The value of $h$ had two effects. First, it reflected the number of sub-networks and thus, the level of communication parallelism. Therefore, a larger $h$ was preferred. Second, for subnetwork types II, III, V and IV, it reflects the level of link contention, so a smaller $h$ value is better. To understand this issue, Fig. 8 (a) compares undirected subnetworks (I, II, and III) when $h = 2$ and 4. Directed subnetworks (IV, V, and VI) are compared in Fig. 8 (b).

E) Effects of Hot-Spot Factors: Fig. 9 shows how the hot-spot factor $p$ affects multicast latency. Fig. 9 (a) shows the source hot-spot case. Since the traffic load we injected in this case was not heavy, the U-torus performed the best for all values
Fig. 6. Multicast latency in an $8 \times 8 \times 8$ torus for various numbers of sources when there were: (a) 160, (b) 256, and (c) 510 destinations ($T_s = 30 \mu\text{sec}$, $T_c = 1 \mu\text{sec}$, and $|M_s| = 32$).

Fig. 7. Multicast latency in an $8 \times 8 \times 8$ torus for various message sizes: (a) 64 sources and 160 destinations, (b) 160 sources and 384 destinations, and (c) 384 sources and 384 destinations ($T_s = 300 \mu\text{sec}$ and $T_c = 1 \mu\text{sec}$).
Fig. 8. Effects of $h$ on multicast latency in a 16 $\times$ 16 $\times$ 16 torus: (a) undirected subnetworks and (b) directed subnetworks ($T_s = 300\mu$sec, $T_c = 1\mu$sec, $|M| = 32$, and number of destinations = 160).

Fig. 9. Effects of the hot-spot factor on multicast latency in an 8 $\times$ 8 $\times$ 8 torus: (a) source hot-spot, (b) destination hot-spot, and (c) source-destination hot-spot ($T_s = 30\mu$sec, $T_c = 1\mu$sec, $|M| = 1024$, number of sources = 64, number of destinations = 64).

of $p$. However, for the destination hot-spot case, Fig. 9 (b) shows that the U-torus worsened as the hot-spot factor $p$ increased. This means that destination hot-spot incurs more contention than the source hot-spot does. A similar trend can be seen in Fig. 9 (c) for the source-destination hot-spot case. From these simulations, we can see that our schemes are less sensitive to the hot-spot factor.

F) Throughput: Fig. 10 shows multicast latency when there are five random sources injecting multicasts into the network with arrival rate $\lambda$ (all 5 nodes together). The number of destination nodes was 80, and the message length was 16 flits. We compared the U-torus and our schemes. The U-torus scheme saturated the network much earlier than ours did.
5. MULTI-NODE MULTICAST IN A 3D MESH

In the case of meshes, we only use undirected subnetworks (types I, II and III). The reason is the reachability problem. The discussion in the previous section for tori can be directly applied, here, except for Phase 2, since DDNs are now (dilated) meshes instead of tori.

5.1 Phase 2: Modified Multicasting in DDNs

To perform a multicast \((r_i, M_i, D')\) on a mesh, one possibility is to use the U-mesh scheme [15]. The other is to use the SPU (Source-Partitioned U-mesh) scheme [11]. SPU works exactly the same as the U-torus scheme [18] by sorting and shifting the destination nodes except that now the underlying network is a mesh. It is shown in [11] that a single multicast will remain congestion-free, and that when there are multiple multicasts, the latency can be reduced significantly. In our simulations, we used the SPU in each sub-network.

5.2 Simulation and Performance Comparison

The same parameters used in the torus simulations were used here. Comparisons were made among the U-mesh, SPU, and our schemes.

A) Effects of the Numbers of Sources and Destinations: Fig. 11(a) shows the results obtained when there were 160 destinations. U-mesh had the highest latency, followed by 2I, 2II, SPU, and III. Again, it is seen that more subnetworks will deliver shorter latency.

Fig. 11(b) and (c) show a comparison of results obtained when there were 256 and 512 destinations, respectively. Similar to the torus case, as the traffic load increased, the advantage of our load-balancing schemes became more evident.

B) Effects of the \(T_s/T_c\) Ratio: Fig. 12 shows results obtained when the same simulations in part A were conducted while changing the startup cost \(T_s\) to 30 \(\mu\)sec. The trend for the results is similar to that in Fig. 11.
Fig. 11. Multicast latency in an $8 \times 8 \times 8$ torus for various numbers of sources when there are: (a) 160, (b) 256, and (c) 510 destinations ($T_s = 300\mu sec$, $T_c = 1\mu sec$, and $|M_i| = 32$).

Fig. 12. Multicast latency in an $8 \times 8 \times 8$ mesh for various numbers of sources when there are: (a) 160, (b) 256, and (c) 510 destinations ($T_s = 30\mu sec$, $T_c = 1\mu sec$, and $|M_i| = 32$).
Fig. 13. Multicast latency in an 8 × 8 × 8 mesh for various message sizes: (a) 64 sources and 160 destinations, (b) 160 sources and 384 destinations, and (c) 384 sources and 384 destinations (\(T_s = 300\ \mu \text{sec}\) and \(T_c = 1\ \mu \text{sec}\)).

C) Effects of the Message Length: Fig. 13 compares the U-mesh and SPU with our schemes by showing results obtained when the message size was changed. Generally, larger messages favor our scheme.

D) Effects of \(h\): As mentioned earlier, the value of \(h\) reflects: (i) the level of communication parallelism and (ii) the level of link contention. Fig. 14 shows that the former plays a more important role.

E) Effects of Hot Spots: Fig. 15 shows the effect of hot spots on latency. Similar to the torus case, this shows that both the U-mesh and SPU schemes are insensitive to source hot spots but are sensitive to destination hot spots and source-destination hot spots. On the contrary, all our schemes are quite insensitive to these hot-spot scenarios. These simulations justify the effectiveness of our load balancing approach.

F) Throughput: Fig. 16 shows the resulting multicast latency when five random sources were injecting multicasts into the network at arrival rate \(\lambda\). The number of destination nodes was 80, and the message length was 16 flits. The U-mesh scheme saturated the network at \(\lambda = 0.015\), while ours did so at \(\lambda = 0.025\). SPU was slightly worse than ours.

6. Subnetworks of Higher Dimensional Tori/Meshes

In this section, we will show how to partition torus/mesh of a higher dimension. An \(n\)-D torus \(T_{e_1e_2...e_n}\) consists of \(e_1e_2...e_n\) nodes, each denoted as \(P_{d_1d_2...d_n}\), where \(0 \leq d_k < e_k, k = 1...n\).
Fig. 14. Effects of $h$ on multicast latency in a $16 \times 16 \times 16$ mesh: ($T_s = 300\mu$sec, $T_c = 1\mu$sec, $|M| = 32$, and number of destinations = 160).

Fig. 15. Effects of the hot-spot factor on multicast latency in an $8 \times 8 \times 8$ mesh: (a) sources hot spot, (b) destinations hot spot, and (c) source-destinations hot spot ($T_s = 30\mu$sec, $T_c = 1\mu$sec, $|M| = 1024$, number of sources = 64, and number of destinations = 64).

Fig. 16. Latency vs. arrival rate in an $8 \times 8 \times 8$ mesh: ($T_s = 30\mu$sec, $T_c = 1\mu$sec, $|M| = 16$, number of sources = 5, number of destinations = 80).
Definition 11 Given an integer $j$, a torus $T_{e_1\times e_2 \times \ldots \times e_n}$, and any integer $h$ that divides each of $e_1$, $e_2$, ..., $e_n$, we define type-$j$ subnetworks as a set of $h^j$ subnetworks $G_{i_1, \ldots, i_j}^j = (V_{i_1, \ldots, i_j}^j, E_{i_1, \ldots, i_j}^j)$, where $i_1, i_2, \ldots, i_j \in \{0, 1, \ldots, h-1\}$, such that

$$V_{i_1, \ldots, i_j}^j = \{ p_{d_1, d_2, \ldots, d_n} \mid d_k = c_k h + i_k \text{ when } 1 \leq k \leq j \text{ and } \frac{d_k}{h} = \frac{i_k}{h} \text{ for } k > j \}
$$

for all $c_k = 0, \frac{e_k}{h} - 1$,

$$E_{i_1, \ldots, i_j}^j = \{ \text{all channels along the } k\text{-th axis such that the axis crosses a node } p_{d_1, d_2, \ldots, d_n} \in V_{i_1, \ldots, i_j}^j \text{ for } k = 1 \ldots n \}.
$$

Lemma 5 The $h^j$ subnetworks $G_{i_1, \ldots, i_j}^j$, $i_1, i_2, \ldots, i_j \in \{0, 1, \ldots, h-1\}$, defined in Definition 11, are free from node contention but have link contention $h^{j-1}$.

It is even possible to enforce a direction on each link in the above subnetworks. This is similar to the earlier 3-D torus case. This will reduce the link contention by half. We will leave this part to the reader.

For the case of meshes, the above definition can still be used except that the wrap-around links should be removed. Again, extension will be straightforward.

7. CONCLUSIONS

In this paper, we have developed a set of efficient schemes for multi-node multicast in a multi-dimensional torus/mesh. This work has successfully extended our earlier work [22] from 2D tori/meshes to higher dimensional ones. 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 throughout the whole network. Extensive simulations have been conducted, which show significant improvement over the existing U-torus, U-mesh, and SPU schemes. Our investigation on hot-spot scenarios also justifies the importance of performing load balancing under heavy traffic loads.

REFERENCES


Ming-Hour Yang (楊明豪) received his MS degree in electronic engineering from the Chung-Hua University, Taiwan, in 1996, and his Ph.D. degree in Computer Science and Information Engineering from the National Central University, Taiwan, in 2001. He was a Research Fellow of National Strategic Studies Institute at National Defense University, Taiwan. He is currently an Associate Professor of the Department of Computer Science and Information Engineering, National Chiao-Tung University, Taiwan. His interests include video on demand, parallel and distributed computing, wireless network, and network security.

Yu-Chee Tseng (曾煜棋) received his B.S. and M.S. degrees in Computer Science from the National Taiwan University and the National Tsing-Hua University in 1985 and 1987, respectively. He worked for the D-LINK Inc. as an engineer in 1990. He obtained his Ph.D. in Computer and Information Science from the Ohio State University in January of 1994. From 1994 to 1996, he was an Associate Professor at the Department of Computer Science, Chung-Hua University. He joined the Department of Computer Science and Information Engineering, National Central University in 1996, and has become a Full Professor since 1999. Since Aug. 2000, he has become a Full Professor at the Department of Computer Science and Information Engineering, National Chiao-Tung University, Taiwan. Dr. Tseng served as a Program Committee Member in the International Conference on Parallel and Distributed Systems, 1996, the International Conference on Parallel Processing, 1998, the International Conference on Distributed Computing Systems, 2000, and the International Conference on Computer Communication and Networks 2000. He was the Chair of the first and second Wireless Networks and Mobile Computing Workshop, in 2000 and 2001. His research interests include wireless communication, network security, parallel and distributed computing, and computer architecture. Dr. Tseng has served as a Guest Editor in journals ACM Wireless Networks, IEEE Transactions on Computers, and Wireless Communications and Mobile Computing. Dr. Tseng is a member of the IEEE Computer Society.

Ming-Shian Jian (簡銘宣) received BS from Tatung University in 1998, MS from National Central University in 2000, in Computer Science and Information Engineering. He worked on parallel and distributed computing, and wireless network. From Jan. 2001 to now, he joined YODA communications incorporation, research and development group, in Science Based Industrial Park Hsinchu, Taiwan, R.O.C..