Complete Exchange in General Multidimensional Mesh Networks*

Young-Joo Suh†, Kang G. Shin†, and Sudhakar Yalamanchili¥

†Real-Time Computing Laboratory
Department of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, MI 48109-2122

E-mail: {yjsuh, kgshin}@eecs.umich.edu
Tel: (313) 763-0391, **Fax:** (313) 763-4617

¥Computer Systems Research Laboratory
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332-0250

E-mail: sudha@ece.gatech.edu
Tel: (404) 894-2940, **Fax:** (404) 894-9959

Abstract

In this paper, we present new algorithms for all-to-all personalized exchange or complete exchange in multidimensional mesh-connected multiprocessors with an arbitrary number of nodes in each dimension. For an $R \times C$ mesh where $R \le C$, the proposed algorithm has time complexities of $O(C)$ message start-ups and $O(RC^2)$ message transmissions. The algorithms for three or higher dimensional meshes follow a similar structure. Unlike existing message combining algorithms in which the number of nodes should be a power-of-two square, the proposed algorithms accommodate meshes with an arbitrary number of nodes in a dimension. In addition, destinations remain fixed over a larger number of steps in the proposed algorithms, thus making them amenable to optimizations. Finally, the data structures used are simple and hence make substantial saving of message-rearrangement time.

1 Introduction

Interprocessor communication may become a main bottleneck to scalable parallel implementations of computationally intensive applications. This has motivated the development of efficient and innovative algorithms for demanding interprocessor communication patterns such as *collective communication* pattern [7]. Among these, *all-to-all personalized exchange* or *complete exchange* is perhaps the most demanding communication patterns [1,3,4,5,8]. In complete exchange, every processor communicates a block of distinct data to every other processor in the system. In an N-node system, each node P_i, $1 \le i \le N$, has N blocks of data, $B[i, 1], B[i, 2], \ldots, B[i, N]$, with a distinct block for each other node. After the exchange operation, each node P_i has N blocks, $B[1, i], B[2, i], \ldots, B[N, i]$, one from each other node. Many scientific parallel algorithms require the complete exchange communication pattern, e.g., matrix transpose operations, fast Fourier transform (FFT), and solutions to the n-body problem.

Several studies by Bokhari and Berryman [2], Sunder et al. [10], and Tseng et al. [13] have produced algorithms using message combining in $2^d \times 2^d$ meshes or tori. These algorithms incur an execution time of $O(2^d)$ due to message start-ups and $O(2^{3d})$ due to message transmissions. Recently, Suh and Yalamanchili [9] proposed algorithms using message combining in $2^d \times 2^d$ and $2^d \times 2^d \times 2^d$ tori with time complexities of $O(d)$ due to message start-ups and $O(2^{3d})$ (in 2D) or $O(2^{4d})$ (in 3D) due to message transmissions. These algorithms differ from each other primarily in the manner in which pairwise exchange operations are scheduled. However, they have all been defined for meshes or tori where the number of processors in each dimension is an integer power-of-two.

In this paper, we present new algorithms for complete exchange for multidimensional meshes with an arbitrary number of nodes in each dimension. The algorithms utilize message combining to reduce the time associated with message start-ups. The proposed algorithms are suitable for a wide range of mesh topologies. This is particularly useful in multi-programmed multiprocessor architectures where, in the interest of efficiency, processor allocation algorithms attempt to maximize processor utilization - not the shape of the allocated sub-mesh. As a result, allocated sub-meshes are rarely square and are not under the control of the application programmer [14]. Furthermore, problem sizes may dictate the use of non-square grid in which case the set of processors allocated to a program normally does not form a square sub-mesh. Moreover, when the proposed algorithms are applied to the special case of "square" 2D meshes where the number of processors in each dimension is a power-of-two, they improve performance over the best existing algorithm for power-of-two square meshes [10].

* The work reported in this paper was supported in part by the National Science Foundation under Grant MIP-9203895.

The salient features of the proposed algorithms are (i) unlike existing message combining algorithms, they accommodate meshes with an arbitrary number of nodes in a dimension, (ii) they are simple in that destinations remain fixed over a larger number of steps, and are thus amenable to optimizations, e.g., caching of message buffers, locality optimizations, etc., (iii) they are the first message combining algorithms for such three or higher dimensional meshes, and (iv) the data structures are simple, thus saving substantial message-rearrangement time.

The following section presents performance model and parameters. We propose the algorithm for 2D meshes in Section 3. The algorithm is extended to three or higher dimensional meshes in Section 4. Section 5 evaluates the performance of the proposed algorithms. The results are discussed in Section 6.

2 Performance Model and Parameters

The target architecture is a mesh-connected, wormhole-switched multiprocessor. Each message is partitioned into a number of *flits*. We assume that each processor has *N* distinct *m*-flit message blocks, the channel is one-flit wide, and each processor has one pair of injection/consumption buffers for the internal processor-router channel (i.e., one-port architecture). All links are full-duplex channels.

A common metric used to evaluate the performance of inter-processor communication is *completion time* or *communication time*. In general, the communication time includes start-up time (t_s), message transmission time per flit (t_c), the propagation delay (t_l), and data rearrangement time between communication steps/phases (ρ). The propagation delay is the per-hop time multiplied by the number of links traversed by a message. It is not negligible in packet-switched networks, but it is negligible in current generation wormhole-switched networks.

In this paper, a *step* is the basic unit of a contention-free communication and a *phase* is a sequence of steps. The completion time for one communication step can be approximated by $T = t_s + m \cdot t_c$ if an *m*-byte message block is transmitted, without any contention, to the destination using wormhole switching. Between successive communication steps or phases for complete exchange, one may need to rearrange data.

3 Algorithms for 2D Meshes

3.1 Node Groups

For an $R \times C$ mesh, where R and C are even numbers and $R \leq C$, each node is identified by a label $P(r, c)$, $0 \leq r \leq R-1$ and $0 \leq c \leq C-1$. Each node is included in one of four node groups according to the following rules.

IF r and c are even, node P is included in group EE.

IF r is odd and c is even, node P is included in group OE.

IF r is even and c is odd, node P is included in group EO.

IF r and c are odd, node P is included in group OO.

The nodes in a group form an $\frac{R}{2} \times \frac{C}{2}$ submesh. For example, in a 6×6 mesh shown in Figure 1, nine nodes with an identical marking form a 3×3 submesh. The proposed algorithm consists of three phases. In phases 1 and 2, messages are transmitted among nodes in the same group. Nodes in each group and in the same row (or column) form a logical ring. For example, in a 6×6 mesh shown in Figures 1(b) and (c), three nodes in each row or column form a logical ring. After phase 2, the network is divided into $\frac{RC}{4}$ contiguous 2×2 submeshes and all of the four nodes in a 2×2 submesh are included in distinct node groups (see Figure 1(d)). In phase 3, message transmissions are performed with nodes in distinct groups and in the same 2×2 submesh to finish the complete exchange operation.

3.2 Communication Pattern

We now describe the communication pattern of the proposed algorithm without considering the data storage structure. The detailed description for the proposed algorithm is presented in the next subsection.

The proposed algorithm consists of three phases. The following operations are performed in phase 1:

Phase 1

IF $P(r, c) \in EE \ or \ OO$,
then P(r,c) transmits to P(r, (c+2)*mod C*)

IF $P(r, c) \in EO \ or \ OE$,
then P(r,c) transmits to P((r+2)*mod R*, c).

Phase 1 requires $\frac{C}{2} - 1$ steps. Throughout the $\frac{C}{2} - 1$ steps of phase 1, each node transmits messages to a fixed destination node - the next node in the logical ring to which the node belongs. Since the size of a submesh is $\frac{R}{2} \times \frac{C}{2}$, there are at most $\frac{C}{2}$ nodes in a row or column (because $R \leq C$). Consider a node A's blocks to be scattered to all nodes. In step 1, node A transmits all of its blocks except those to be transmitted by itself in phases 2 and 3 to the next node (e.g., node B) in the logical ring. Node B extracts blocks to be transmitted by itself in phases 2 and 3, then transmits the remaining blocks to the next node (e.g., node C) in the logical ring in step 2. This procedure repeats itself and in the last step of phase 1, the last node (e.g., node L) in the logical ring receives only the blocks to be transmitted by the node in phases 2 and 3. Likewise, the other nodes in the logical ring and all nodes in the other logical rings also scatter their blocks to all nodes in their logical rings. If $R \neq C$, then nodes in groups EO and OE finish the operations in phase 1 in $\frac{R}{2} - 1$ steps and become idle during the remaining $\frac{C-R}{2}$ steps.

In phase 2, each node changes dimensions then performs a cyclic shift operation along a logical ring in the new dimension. In phase 2, the following operations are performed:

Phase 2

IF $P(r, c) \in EE \ or \ OO$,
then P(r,c) transmits to P((r+2)*mod R, c*).

IF $P(r, c) \in EO \ or \ OE$,
then P(r,c) transmits to P(r, (c+2)*mod C*).

Phase 2 also requires $\frac{C}{2} - 1$ steps and the communication pattern is the same in the new dimension as that in phase 1. Each node in a logical ring of phase 1 (e.g., each node A, B, C, ..., L) transmits blocks along logical rings in the new dimension in parallel. In each step, each node extracts blocks for itself and blocks to be transmitted by itself in phase 3, then transmits the remaining blocks to the next node in the logical ring. Thus, after $\frac{C}{2} - 1$ steps of phase 2, each node has blocks originated from nodes in the same node group, destined for itself and to be transmitted by the node in phase 3. As in phase 1, if $R \neq C$ then nodes in groups EE and OO finish the operations in phase 1 in $\frac{R}{2} - 1$ steps and idle during the remaining $\frac{C-R}{2}$ steps.

Now, the network can be divided into $\frac{RC}{4}$ contiguous 2×2 submeshes. All nodes in a 2×2 submesh are included in distinct node groups and have blocks originated from all nodes in their respective groups. In phase 3, two steps are required as follows:

Phase 3 Step 1

IF $P(r, c) \in EE \ or \ OE$, then P(r,c) transmits to P(r, c+1).

IF $P(r, c) \in OO \ or \ EO$, then P(r,c) transmits to P(r, c-1).

Phase 3 Step 2

IF $P(r, c) \in EE \ or \ OE$, then P(r,c) transmits to P(r+1, c).

IF $P(r, c) \in OO \ or \ EO$, then P(r,c) transmits to P(r-1, c).

After phase 3, each node has RC blocks originated from all nodes in the network to complete the complete exchange operation. The communication pattern of the proposed algorithms in a 6×6 mesh is captured in Figure 1.

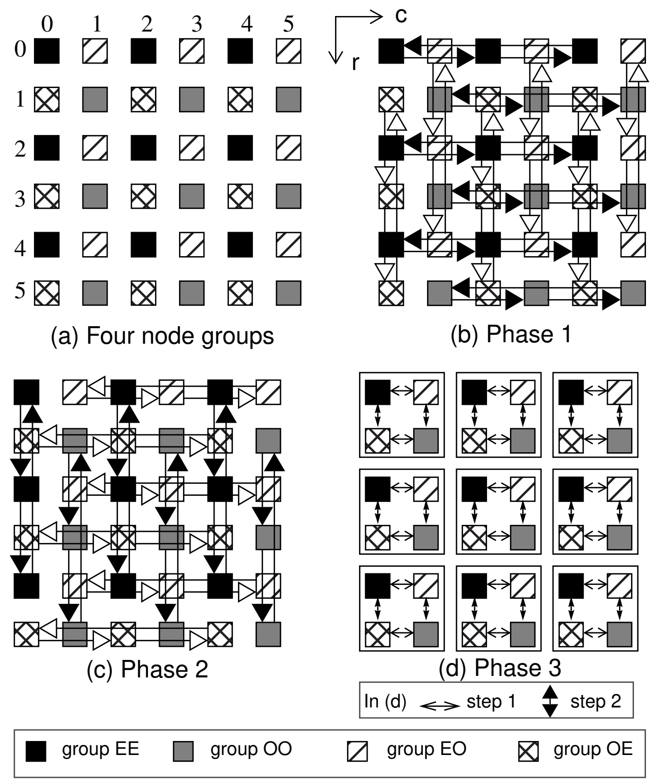

Figure 1. Communication pattern in a 6x6 mesh.

3.3 Data Array

In this subsection, the contents of transmitted blocks and array structure in each communication step are described in detail.

Initially, each node has RC distinct blocks to distribute to other nodes in two dimensional array B[u,v], where $0 \leq u \leq R-1$ and $0 \leq v \leq C-1$ if the node is included in groups EE or OO, and $0 \leq u \leq C-1$ and $0 \leq v \leq R-1$ if the node is included in groups EO or OE. We assume that the array is ordered in column major, and if blocks to be transmitted are not contiguous, then they should be rearranged before the transmission. The initial data structure of a node is dependent upon the communication pattern in phases 1 and 2. In B[u,0], a block destined for the node that is u hops away from the node along the direction of the logical ring in phase 1 (including nodes not in the logical ring) is located. In B[u,v], a block destined for the node that is v hops away from the node for which the block B[u,0] is destined along the direction of the logical ring in phase 2 (including nodes not in

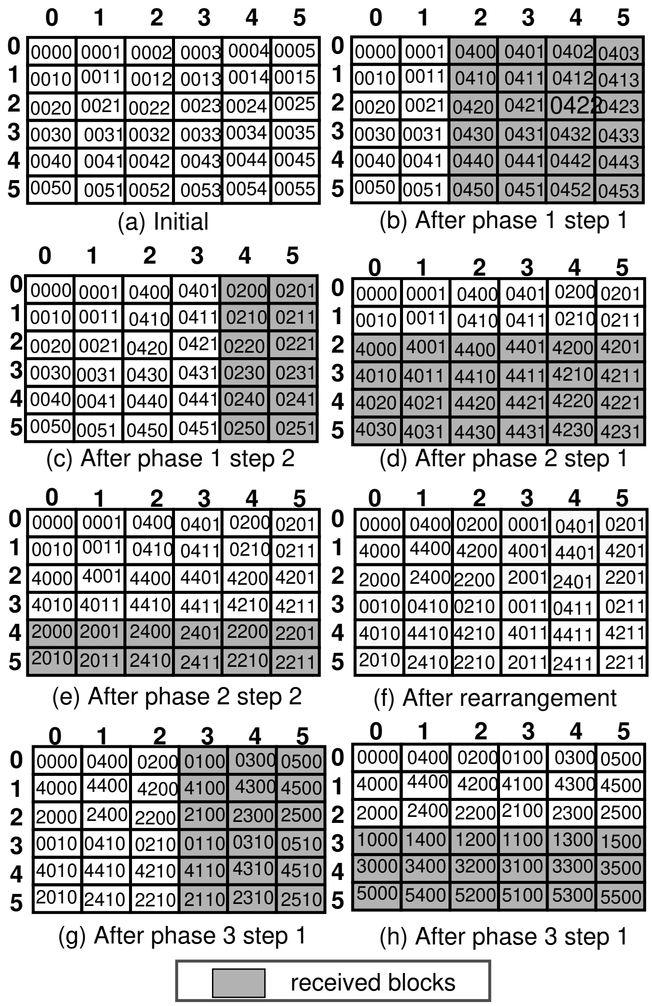

Figure 2. Logical data structure in node (0,0) for complete exchange in a 6x6 mesh.

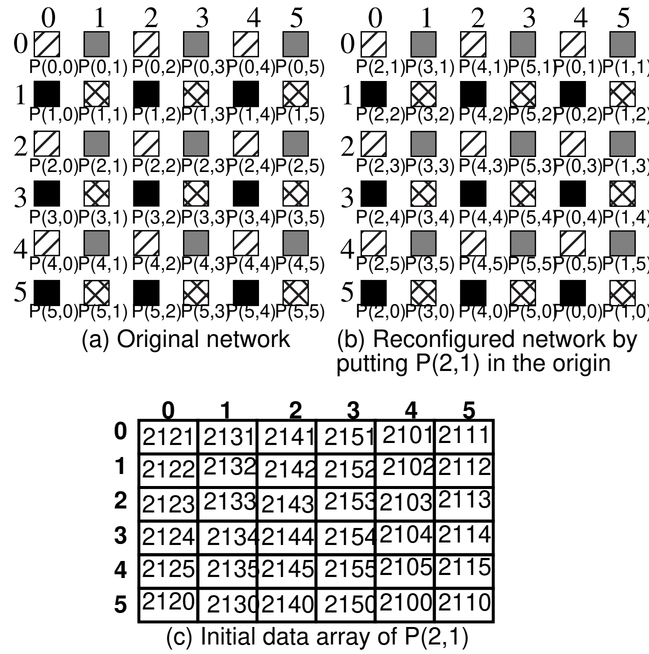

Figure 3. Reconfigured network and initial data array of P(2,1).

the logical ring) is located. For example, consider the initial data array of node P(0,0) for a 6×6 mesh illustrated in Figure 1. It is included in group EE and nodes in the group are assigned the positive column and positive row directions in phases 1 and 2, respectively. Thus, the first row of the data array includes blocks destined for nodes in the same row along the positive row direction from P(0,0) in that order, and the other cells in each column of the data array include blocks destined for nodes in the same column along the positive column direction from each node in the first row, in that order, as illustrated in Figure 2(a). It is exactly the same configuration as the network itself. It may be easier to understand the initial data structure in each node as follows: The network is reconfigured by placing each node in the origin (0,0) and by making its message transmission operation in phase 1 to be performed along the row. Figure 3(b) illustrates the reconfigured network by placing the node P(2,1) at the origin and making its message transmission operation in phase 1 to be performed along the row. Then, the initial data structure follows exactly the same configuration as that of the reconfigured network as illustrated in Figure 3(c). In the figure, a block is identified by the combination of source node ID and destination node ID. For example, a block in node P(1,2) which is destined for node P(3,4) is identified by block 1234.

In step i, $1 \le i \le \frac{C}{2} - 1$, of phase 1, each node transmits blocks in columns $2i$ through $C-1$, i.e., B[*,2i..C-1] (where '*' is the wild card designation), to its destination node in phase 1, while receiving the same number of blocks from its source node in phase 1: In step 1, each node transmits all blocks except blocks that can be transmitted by itself in phases 2 and 3 (i.e., blocks in the first two columns). Among the received blocks in step 1, each node extracts the blocks that can be transmitted by itself in phases 2 and 3 (i.e., blocks in the third and fourth columns), then transmits the remaining blocks to its destination node in step 2. This procedure repeats until the last step of phase 1. After phase 1, each node has blocks from all nodes in the same logical ring of phase 1, destined for nodes that the node can transmit in the remaining phases.

In step j, $1 \le j \le \frac{C}{2} - 1$, of phase 2, each node transmits blocks in rows $2j$ through $C-1$, i.e., B[2j..C-1,*] to its destination node in phase 2, while receiving the same number of blocks from its source node in phase 2: In step 1, each node transmits all blocks except blocks that can be transmitted by itself in phase 3 (i.e., blocks in the first two rows). Among the received blocks in step 1, each node extracts the blocks that to be transmitted by itself in phase 3 (i.e., blocks in the third and fourth rows) then transmits the remaining blocks to its destination node in step 2. This procedure repeats until the last step of phase 2.

After phase 2, each node in a 2×2 submesh has RC blocks originated from all nodes in the same node group destined for four nodes in the 2×2 submesh to which the node belongs. But blocks destined for each node in the 2×2 submesh are distributed. Thus, before phase 3, the blocks are rearranged: In the upper left quadrant, blocks destined for itself are placed and blocks destined for the destination node in step 1, blocks destined for the node in the diagonal in the 2×2 submesh, and blocks destined for the destination node in step 2 are located in other quadrants, in clock-wise order. In step 1 of phase 3, each node transmits $\frac{RC}{2}$ blocks destined for the two nodes in the other column of the 2×2 submesh, while receiving the same number of blocks from the destination node. After step 1, each node has $\frac{RC}{2}$ blocks destined for itself and another $\frac{RC}{2}$ blocks destined for the destination node in step 2. Thus, the latter $\frac{RC}{2}$ blocks are transmitted in step 2, while receiving the same number of blocks. Now, every node has RC blocks, one block from every node in the network.

Consider an example in a 6×6 mesh illustrated in Figure 1. The data array of P(0,0) in each communication step is captured in Figure 2. Initial data array of P(0,0) is shown in Figure 2(a). The communication pattern in phase 1 is illustrated in Figure 1(b). In phase 1 step 1, P(0,0) sends blocks in columns 2 through 5 to P(0,2) while receiving the same number of blocks from P(0,4) (Figure 2(b)). In step 2 of phase 1, blocks in the last two columns are transmitted (Figure 2(c)). Now, phase 2 initiates. The communication pattern in phase 2 is illustrated in Figure 1(c). In step 1 of phase 2, blocks in rows 2 through 5 are transmitted to P(2,0), while receiving blocks from P(4,0) (Figure 2(d)). In step 2 of phase 2, blocks in the last two rows are transmitted to P(2,0), while receiving blocks from P(4,0). After phase 2, all blocks in a node are those destined for nodes in the same 2×2 submesh (i.e., P(0,0), P(0,1), P(1,0), and P(1,1)) as shown in Figure 2(e). But blocks destined for each node in the 2×2 submesh are distributed. Thus, before phase 3, the blocks are rearranged as shown in Figure 2(f). Now, phase 3 starts and the communication pattern in phase 3 is illustrated in Figure 1(d). The data array after each step of phase 3 are captured in Figures 2(g) and (h).

3.4 Complexity Analysis

In this subsection, we analyze the time cost required by the proposed algorithm in terms of start-up cost, message transmission cost, and data rearrangement cost.

(a) Start-up cost: For an $R \times C$ 2D mesh, $R \leq C$, there are $\frac{C}{2} - 1$ steps in phase 1, $\frac{C}{2} - 1$ steps in phase 2, and 2 steps in phase 3. Thus a total of C steps are required.

(b) Message transmission cost: In step p of phase 1, where $1 \leq p \leq \frac{C}{2} - 1$, $max\{R(C-2p), C(R-2p)\}$, i.e., $R(C-2p)$ blocks (since $R \leq C$) are transmitted. In step q of phase 2, where $1 \leq q \leq \frac{C}{2} - 1$, $R(C-2q)$ blocks are transmitted. In phase 3, there are two steps and $\frac{RC}{2}$ blocks are transmitted in each step. Thus, the total number of transmitted blocks is $2 \cdot \sum_{p=1}^{\frac{C}{2}-1} R(C-2p) + 2\left(\frac{RC}{2}\right) = \frac{RC^2}{2}$.

(c) Data rearrangement cost: In the logical 2-dimensional data array described in the previous subsection, only one data rearrangement step is required before phase 3 (see Figure 2(f)). But, in the physical model, the array is ordered in column major, and if blocks to be transmitted are not contiguous, then they should be rearranged before the transmission. Blocks should be rearranged between two consecutive phases. Since there are three phases, two message-rearrangement steps are required. The total data rearrangement cost is $2(RC)m\rho$.

4 Extension to *n*-Dimensional Meshes

The algorithm in 2D meshes can also be extended to $a_1 \times a_2 \times \ldots \times a_n$ n-dimensional meshes in a straightforward manner.

For an $a_1 \times a_2 \times \ldots \times a_n$ n-dimensional mesh, there are $n+1$ phases. Each node is included in one of 2^n node groups according to whether each of the coordinates is even or odd. The nodes in a group form an $\frac{a_1}{2} \times \ldots \times \frac{a_n}{2}$ n-dimensional submesh. In each of the first n phases, messages are transmitted along logical rings in each dimension. In the last phase (phase $n+1$), message exchange operations are performed in each $2 \times 2 \times \ldots \times 2$ n-dimensional submesh. To avoid channel contentions, the dimensions in which messages are transmitted are distributed in each phase.

In general, for n-dimensional networks, nodes in the even numbered unit along the dimension n follow the communication patterns of (n-1)-dimensional networks in the first n-1 phases and then perform the communication along the last dimension (i.e., dimension n) in phase n, while the other nodes perform the communication along the dimension n in phase 1 and then follow the communication patterns of (n-1)-dimensional networks in the remaining n-1 phases. Finally, in phase n+1, message exchange operations are performed in each $2 \times 2 \times \ldots \times 2$ n-dimensional submesh and there are n steps.

The data array in each node is very similar to that in 2D meshes. Figure 4 illustrates the transmitted blocks in each step until phase 3 for a $6 \times 6 \times 6$ 3D mesh. After phase 3, each node in a $2 \times 2 \times 2$ submesh has a^3 blocks originated from all nodes in the same node group ($\frac{a^3}{8}$ nodes) destined for eight nodes in the $2 \times 2 \times 2$ submesh to which the node belongs. But blocks destined for each node in the $2 \times 2 \times 2$ submesh are distributed. Thus, before phase 4, the blocks are rearranged. In each step phase 4, each node transmits $\frac{a^3}{2}$

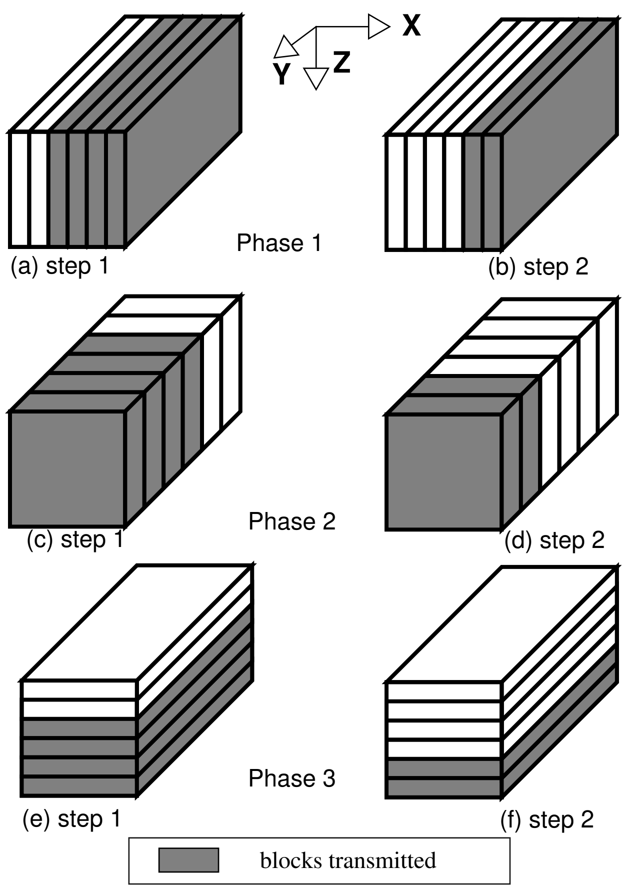

Figure 4. Message blocks transmitted in each step in a 6x6x6 mesh.

blocks.

In general, for an $a_1 \times a_2 \times \ldots \times a_n$ n-dimensional mesh, where $a_1 \geq a_2 \geq \ldots \geq a_n$, there are n+1 phases. In the first n phases, $\frac{a_1}{2} - 1$ steps are required per phase and, in phase n+1, n steps are required. Thus, the total number of steps required for the complete exchange operation is $\frac{n}{2}a_1$. In step s, $1 \leq s \leq \frac{a_1}{2} - 1$, in each of the first n phases, $max\{(a_1 - 2 \cdot s)(a_2 a_3 \ldots a_n), \ldots, (a_n - 2 \cdot s)(a_1 a_3 \ldots a_{n-1})\}$ i.e., $(a_1 - 2 \cdot s)(a_2 \ldots a_n)$ blocks are transmitted (since $a_1 \geq a_2 \geq \ldots \geq a_n$). In each step of phase n+1, $\frac{1}{2}(a_1 a_2 \ldots a_n)$ blocks are transmitted. Thus, the total number of transmitted blocks is as follows:

$$n \sum_{s=1}^{\frac{a_1}{2}-1} (a_1 - 2 \cdot s)(a_2 \ldots a_n) + \frac{n}{2}(a_1 a_2 \ldots a_n) = \frac{n}{4}(a_1^2 a_2 \ldots a_n).$$

Between each of two successive phases, blocks are rearranged to prepare for the next phase. Since there are n+1 phases, n message-rearrangement steps are required. That is, the total data rearrangement cost is $n(a_1 a_2 \ldots a_n)m\rho$.

5 Performance Evaluation

In this section, the performance of the proposed algorithms is evaluated and compared with that of existing algorithms. In previous sections, we analyzed the time cost of the proposed algorithm in terms of three dominant terms in completion time: start-up cost, message transmission cost, and data rearrangement cost.

Network	Start-up Time	Message Transmission Time	Data Rearrangement Time
$R \times C$ 2D mesh	$(C)t_s$	$\left(\frac{RC^2}{2}\right)mt_c$	$2(RC)m\rho$
$a_1 \times \ldots \times a_n$ nD mesh	$\left(\frac{n}{2}a_1\right)t_s$	$\frac{n}{4}(a_1^2 a_2 \ldots a_n)mt_c$	$n(a_1 a_2 \ldots a_n)m\rho$

Table 1: Performance summary of proposed algorithms.

The asymptotic time complexities of the proposed algorithms are summarized in Table 1. In 2D meshes, the total number of required communication steps is the length of the longer dimension. In general, in an $a_1 \times a_2 \times \ldots \times a_n$ n-dimensional mesh, where $a_1 \geq a_2 \geq \ldots \geq a_n$, the time complexity of the start-up cost is $O(a_1)$ and that of message transmission cost is $O(a_1^2 a_2 \ldots a_n)$. In addition, n message-rearrangement steps are required.

The time complexities of existing algorithms for complete exchange in a $2^d \times 2^d$ mesh using message combining [2,10] are summarized in Table 2. The time complexities of the proposed 2D algorithm applied to power-of-two square meshes are also summarized in Table 2. As shown in the table, the time complexities of the start-up time and message transmission time are equivalent to those proposed in [10]. But, the proposed algorithm is advantageous with respect to the data rearrangement time. Data rearrangement time is required between phases to prepare for the next phase. In a $2^d \times 2^d$ mesh, there are d phases in the algorithms [2] and [10], thus $d-1$ rearrangement steps are required. However, the proposed algorithm requires only two rearrangement steps since there are three phases. Thus, overall effect is that the proposed algorithm exhibits better performance. At the very least, it appears the algorithms are comparable, and the proposed algorithms possess other advantages: the size of each dimension need not be power-of-two square, extensions to multi-dimensional meshes.

Network	Start-up Time	Message Transmission Time	Data Rearrangement Time
[2]	$3(2^d-1)t_s$	$3(2^d-1)2^{2d-2}mt_c$	$(d-1)2^{2d}m\rho$
[10]	$(2^d)t_s$	$(2^{3d-1})mt_c$	$(d-1)2^{2d}m\rho$
Proposed	$(2^d)t_s$	$(2^{3d-1})mt_c$	$(2)2^{2d}m\rho$

Table 2: Comparison of algorithms.

6 Conclusions

This paper has proposed new algorithms for complete exchange for multidimensional mesh-connected networks. Although the algorithms targeted wormhole-switched networks, they can be efficiently used in virtual cut-through or circuit-switched networks. The proposed algorithms utilize message combining to reduce the time complexity of message start-ups. Unlike existing message combining algorithms, the proposed algorithms accommodate non-power-of-two networks of arbitrary dimensions. It is interesting that, when the algorithms are applied to power-of-two square meshes, the proposed algorithms exhibit better performance than existing algorithms, suggesting their utility in various multiprocessing environments without performance degradation.

References

[1] S. H. Bokhari, Multiphase Complete Exchange on Paragon, SP2, and CS-2, *IEEE Parallel & Distributed Technology*, pages 45-59, Fall 1996.

[2] S. H. Bokhari and H. Berryman, Complete Exchange on a Circuit Switched Mesh, *Proceedings of Scalable High Performance Computing Conference*, pages 300-306, 1992.

[3] J. Bruck, C. T. Ho, S. Kipnis, and D. Weathersby, Efficient Algorithms for All-to-All Communications in Multi-Port Message-Passing Systems, *Proceedings of Symposium on Parallel Algorithms and Architectures*, pages 298-309, 1994.

[4] S. Hinrichs, C. Kosak, D. R. O'Hallaron, T. M. Sticker, and R. Take, An Architecture for Optimal All-to-All Personalized Communication, *Proceedings of Symposium on Parallel Algorithms and Architectures*, pages 310-319, 1994.

[5] S. L. Johnsson and C. T. Ho, Optimum Broadcasting and Personalized Communication in Hypercubes, *IEEE Trans. on Computers*, vol. 38, no. 9, pages 1249-1268, Sep. 1989.

[6] V. Kumar, A. Grama, A. Gupta, and G. Karypis, Introduction to Parallel Computing - Design and Analysis of Algorithms, Benjamin/Cummings Publishing Company, chapter 3, 1994.

[7] P. K. McKinley, Y.-J. Tsai, and D. F. Robinson, A Survey of Collective Communication in Wormhole-Routed Massively Parallel Computers, *Tech. Report MSU-CPS-94-35*, Michigan State University, 1995.

[8] D. S. Scott, Efficient All-to-All Communication Patterns in Hypercube and Mesh Topologies, *Proceedings of 6th Conference. Distributed Memory Concurrent Computers*, pages 398-403, 1991.

[9] Y. J. Suh and S. Yalamanchili, Algorithms for All-to-All Personalized Exchange in 2D and 3D Tori, *Proceedings of 10th International Parallel Processing Symposium*, pages 815-821, April, 1996.

[10] N. S. Sundar, D. N. Jayasimha, D. K. Panda, and P. Sadayappan, Complete Exchange in 2D Meshes, *Proceedings of Scalable High Performance Computing Conference*, pages 406-413, 1994.

[11] R. Thakur and A. Choudhary, All-to-All Communication on Meshes with Wormhole Routing, *Proceedings of 8th International Parallel Processing Symposium*, pages 561-565, Apr., 1994.

[12] Y.-C. Tseng and S. Gupta, All-to-All Personalized Communication in a Wormhole-Routed Torus, *Proceedings of International Conference on Parallel Processing*, volume 1, pages 76-79, 1995.

[13] Y.-C. Tseng, S. Gupta, and D. Panda, An Efficient Scheme for Complete Exchange in 2D Tori, *Proceedings of International Parallel Processing Symposium*, pages 532-536, 1995.

[14] Cray T3D, *System Architecture Overview*, 1994.