# Configurable Complete Exchanges in 2D Torus-Connected Networks*

Young-Joo Suh
Dept. of Computer Science and Engineering
Pohang University of Science and Technology
San 31, Hyoja-Dong
Pohang 790-784, Korea
yjsuh@postech.edu

Kang G. Shin
Real-Time Computing Laboratory
Department of EECS
The University of Michigan
Ann Arbor, MI 48109-2122
kgshin@eecs.umich.edu

Syungog An
Dept. of Computer Engineering
Paichai University
439-6 Doma-2-Dong
Taejeon 302-735, Korea
sungohk@mail.paichai.ac.kr

## Abstract

*Complete exchange communications are found necessary in many important parallel algorithms. This paper presents algorithms for complete exchange for 2D torus-connected multiprocessors. The proposed algorithms are unique in that they are configurable while trading the time for message startups against larger message sizes. At one extreme, the algorithm minimizes the number of message startups at the expense of increased message-transmission time. At the other extreme, the message-transmission time is reduced at the expense of increased number of message startups. The algorithms are structured such that intermediate solutions are feasible, i.e., the number of message startups can be increased slightly and the message-transmission time is correspondingly reduced. The ability to configure these algorithms makes the proposed algorithms distinct from others and leads to efficient portable implementation of complete exchange algorithms.*

## 1. Introduction

In distributed memory multicomputer systems, it is often required that each processor communicates its data with all other processors. The *all-to-all personalized* or *complete exchange* is one of the most demanding communication patterns in parallel computing. In this communication pattern, every processor communicates a block of distinct data to every other processor in the system [2,3,5,6]. Many scientific applications require the all-to-all personalized exchange communication pattern.

Several studies by Bokhari and Berryman [1], Sunder *et al.* [16] and Tseng *et al.* [19] have produced algorithms using message combining in $2^d \times 2^d$ meshes or tori. These algorithms incur an $O(2^d)$ execution time due to message start-ups and $O(2^{3d})$ time due to message transmissions. Recently, Suh and Yalamanchili [13] proposed algorithms using message combining in $2^d \times 2^d$ and $2^d \times 2^d \times 2^d$ tori having time complexities of $O(d)$ due to message start-ups and $O(2^{3d})$ (in 2D) or $O(2^{4d})$ (in 3D) time due to message transmissions.

This paper presents a set of configurable algorithms for complete exchange for two-dimensional torus-connected networks. The salient feature of the proposed algorithms is that they can be tuned to trade the overheads of message initiation or start-ups against message-transmission time. At one extreme, the algorithm minimizes the number of message start-ups at the expense of increased message-transmission time. At the other extreme, the message-transmission time is significantly reduced at the expense of increased number of message start-ups. The algorithm is structured to yield intermediate solutions, i.e., the number of message start-ups can be slightly increased and the message- transmission time reduced accordingly. The ability to configure these algorithms allows us to match the algorithm characteristics with machine characteristics based on message-initiation overhead and link speeds, to minimize overall execution time. In effect the algorithms can be configured to strike a balance between direct and message-combining approaches on a specific architecture for a given problem size.

* This research was supported in part by Pohang University of Science and Technology, Korea, 1998.

## 2. Performance Model and Parameters

Our target architecture is a torus-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. We also assume that 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.

We will focus on the two dominant components of message latency: start-up time ($t_s$) and message-transmission time ($t_c$). In our model, a step is the basic unit of contention-free communication, i.e., in one-time step, a set of nodes can communicate via disjoint network paths. The duration of a step is determined by the message size and, for large messages, is insensitive to the distance between communicating nodes. The number of steps corresponds to the number of message start-ups. A phase is a sequence of multiple steps.

## 3. Two Algorithms

We now summarize two algorithms (T1 and T2) proposed in [13]. They are message-combining algorithms with a bottom-up approach. The communication proceeds as a number of phases and, within each phase, the algorithms differ in the number of steps. The two algorithms are combined to construct a set of configurable algorithms for 2D tori in Section 4.

### 3.1 Algorithm T1

#### Special Node Groups

For a $2^d \times 2^d$ torus, in exchange phase $i$, $1 \le i \le d$, the communication steps are performed within a $2^i \times 2^i$ submesh or torus. In a $2^p \times 2^p$ submesh, $2 \le p \le d$, we identify two special sets of nodes. The first special group (SG) of nodes in phase $p$, $SG_p(1)$, is defined as the set of nodes along the two main diagonals. The second SG ($SG_p(2)$) is the set of nodes along the main diagonals of the four quadrants *excluding* the elements already in the first SG. If each node is labeled $P(x, y)$, $0 \le x, y \le 2^d - 1$, we can formally define the two groups, $SG_p(1)$ and $SG_p(2)$, as follows.

$P(x, y) \in SG_p(1)$

iff $(y - x) mod 2^p$ OR $(x + y - 2^p - 1) mod 2^p$.

$P(x, y) \in SG_p(2)$

iff $(y - x - 2^{p-1}) mod 2^p$ OR $(x + y - 2^{p-1} - 1) mod 2^p$.

An important property of SGs is that the nodes in the first SG are partitioned into two SGs in the next phase. That is, $SG_j(1) \equiv SG_{j+1}(1) \cup SG_{j+1}(2)$, where $2 \le j \le d-2$.

**Communication Pattern:**

For a $2^d \times 2^d$ torus, the algorithm consists of $d$ exchange phases. Each exchange phase consists of exactly two steps for a total of $2d$ steps. If $d \ge 4$, there is an additional send phase.

In phase 1, message exchanges are performed on each $2 \times 2$ submesh. In this phase, all nodes in the submesh belong to the first SG. In step 1 (step 2) of phase 1, each node sends a block of message to a node whose address is complemented in the least significant bit of the X-coordinate (Y-coordinate). This is represented as follows.

Phase 1 Step 1:

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \to P(x_{d-1} \cdots x_1 \overline{x_0}, y_{d-1} \cdots y_0)$.

Phase 1 Step 2:

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \to P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_1 \overline{y_0})$.

Starting with phase 2 until phase $d-1$, each node in the first SG sends blocks horizontally, while each node in the second SG sends blocks vertically in the first step. In the second step of a phase, each node in the SGs changes dimensions and sends blocks along the new dimension. The message transmissions in two steps of phase $p$, $2 \le p \le d-1$, are summarized as follows.

Phase $p$ Step 1:

$If\ (iam \in SG_p(1))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \to P(x_{d-1} \cdots x_p \overline{x_{p-1} \cdots x_0}, y_{d-1} \cdots y_0)$.

$If\ (iam \in SG_p(2))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \to P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_p \overline{y_{p-1} \cdots y_0})$.

Phase $p$ Step 2:

$If\ (iam \in SG_p(1))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \to P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_p \overline{y_{p-1} \cdots y_0})$.

$If\ (iam \in SG_p(2))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \to P(x_{d-1} \cdots x_p \overline{x_{p-1} \cdots x_0}, y_{d-1} \cdots y_0)$.

where *iam* indicates an arbitrary node $P(x_{d-1} \ldots x_0, y_{d-1} \ldots y_0)$

In phase $d$, the nodes in SGs in phase $d-1$ are also active. The following operations are performed in phase $d$.

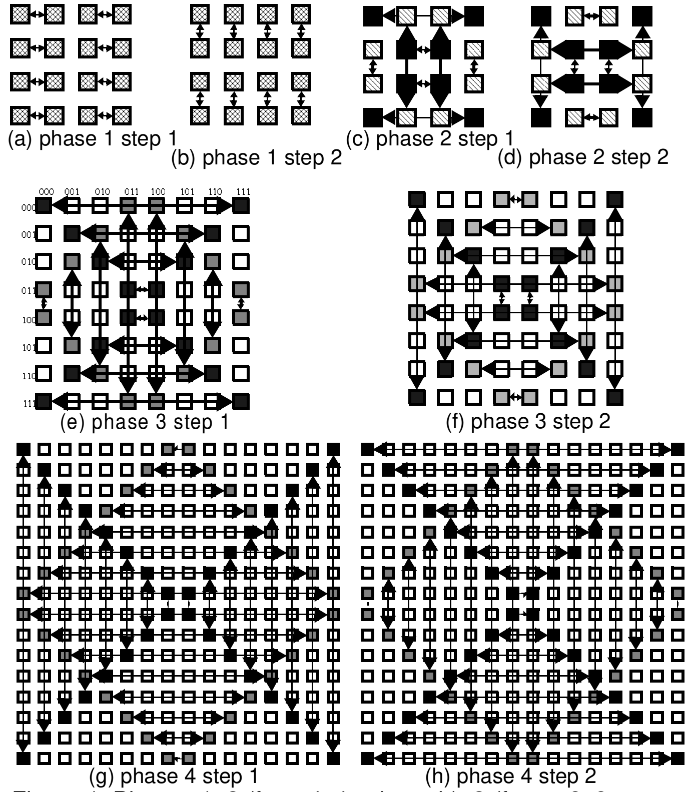



Figure 1. Phases 1, 2 (for a 4x4 submesh), 3 (for an 8x8 submesh), and 4 (for a 16x16 submesh) in a 32x32 torus.

Phase $d$ Step 1:

$If\ (iam \in SG_{d-1}(1))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \to P(\overline{x_{d-1} \cdots x_0}, y_{d-1} \cdots y_0)$.

$If\ (iam \in SG_{d-1}(2))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \to P(x_{d-1} \cdots x_0, \overline{y_{d-1} \cdots y_0})$.

Phase $d$ Step 2:

$If\ (iam \in SG_{d-1}(1))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \to P(x_{d-1} \cdots x_0, \overline{y_{d-1} \cdots y_0})$.

$If\ (iam \in SG_{d-1}(2))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \to P(\overline{x_{d-1} \cdots x_0}, y_{d-1} \cdots y_0)$.

If $d \ge 4$, there is an additional send phase consisting of $d-3$ steps. After phase $d$, each node in $SG_{d-1}(1)$ or $SG_{d-1}(2)$, i.e., each node in $SG_{d-2}(1)$, has all blocks from every other node, each node in $SG_{d-2}(2)$ has blocks from $2^{2(d-1)}$ nodes, each node in $SG_{d-3}(2)$ has blocks from $2^{2(d-2)}$ nodes, and so on. In step $s$ of the send phase, $1 \le s \le d-3$, the following operation is performed.

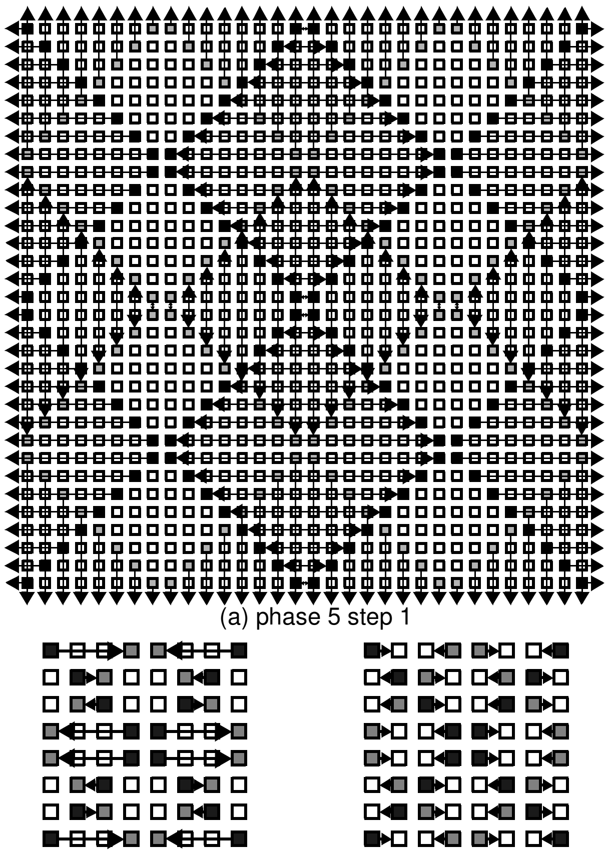
Send phase Step $s$:

$If\ (iam \in SG_{d-s-1}(1))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \to P(x_{d-1} \cdots x_{d-s-2} \overline{x_{d-s-3} \cdots x_0}, y_{d-1} \cdots y_0)$

**Examples:**

Consider a $32 \times 32$ torus. The steps in phases 1 and 2 are shown in Figures 1(a)-(d) for one $4 \times 4$ submesh (the remaining submeshes are identical). In phase 3 (or 4), mes-



(a) phase 5 step 1

(b) send phase step 1 (c) send phase step 2

Figure 2. Phase 5 step 1 and two steps in the send phase (for an 8x8 submesh) in a 32x32 torus.

sage exchange operations are performed within each $8 \times 8$ (or $16 \times 16$) submesh as illustrated in Figures 1(e) and (f) (or Figures 1 (g) and (h)). After phase 4, the nodes in $SG_4(1)$ or $SG_4(2)$ have blocks from all nodes in the quadrant in which those are located. Figure 2(a) shows the first step in phase 5. As shown in the figure, only the active nodes in phase 4 (marked nodes in Figure 2(a), i.e., nodes in $SG_4(1)$ or $SG_4(2)$) participate in exchanging messages.

In the next step, the active nodes change dimensions and exchange messages. Now, each node in $SG_4(1)$ or $SG_4(2)$ (i.e., $SG_3(1)$) has all blocks from all nodes in the torus.

Starting with phase 3, we note that for subsequently larger size submeshes, all processors are not active, i.e., some processors are not members of either SG. The solution is to require one additional phase referred to as a send phase after $d$ exchange phases. In two steps in the send phase, the blocks for non-active nodes are sent as shown in Figures 2(b) and (c), where only one $8 \times 8$ submesh is shown (the remaining submeshes are identical).

**Complexity Analysis:**

The number of steps is $2d$ steps if $d < 4$ and $3d-3$ steps if $d \geq 4$. If $d < 4$, $2^{2d-1}$ blocks are exchanged in each step. Since there are $2d$ steps, the total message transmission time is $(d \cdot 2^{2d})mt_c$. If $d \geq 4$, the number of exchanged blocks per step is not always identical. The total message transmission time is $\{9 \cdot 2^{3d-4} + (d^2 - 5d + 3)2^{2d-1}\}mt_c$.

## 3.2 Algorithm T2

Algorithm T1 focused on minimizing the number of phases (i.e., message start-ups). Algorithm T2 requires more communication steps, but it is simpler and has lower transmission times. The primary distinguishing feature of T2 is that it doesn't have any send phase.

In T1, each exchange phase comprises exactly two steps. For larger size submeshes, not all processors can participate, hence needing a send phase. In T2, each phase is extended to ensure participation of all processors within a phase by identifying additional node groups that are active in the additional steps in a phase.

**Node Groups:**

Groups of nodes (Gs) are defined for Algorithm T2, where the Gs are defined for phases 2 to $d-1$. The Gs in phase $d-1$ are also used in phase $d$. Until phase 2, the Gs in T2 are the same as the SGs in T1. In phase $p$, $3 \leq p \leq d-1$, there are $2^{p-1}$ Gs as follows:

$P(x, y) \in G_p(1)$

iff $(y - x) mod 2^p$ OR $(x + y - 2^p - 1) mod 2^p$.

$P(x, y) \in G_p(2)$

iff $(y - x - 2^{p-1}) mod 2^p$ OR $(x + y - 2^{p-1} - 1) mod 2^p$.

$P(x, y) \in G_p(2k+1)$

iff $\{\{(y - x - 2k) mod 2^p$ OR $(x + y - 2^p - 2k - 1) mod 2^p\}$ AND $(y) mod 2 - 0\}$ OR $\{\{(y - x - 2^p + 2k) mod 2^p$ OR $(x + y - 2k - 1) mod 2^p\}$ AND $(y) mod 2 - 1\}$.

$P(x, y) \in G_p(2k+2)$

iff $\{\{(y - x - 2k) mod 2^p$ OR $(x + y - 2^p - 2k - 1) mod 2^p\}$ AND $(y) mod 2 - 1\}$ OR $\{\{(y - x - 2^p + 2k) mod 2^p$ OR $(x + y - 2k - 1) mod 2^p\}$ AND $(y) mod 2 - 0\}$.

where $1 \leq k \leq 2^{p-2} - 1$. Note that the first two Gs in each phase are the same as the two SGs in T1.

**Communication Pattern:**

For $d < 4$, T2 is identical to T1. For $d \geq 4$, Algorithm T2 consists of $d$ exchange phases but no send phase. The following two steps are performed in phase 1.

Phase 1 Step 1:

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \to P(x_{d-1} \cdots x_1 \overline{x_0}, y_{d-1} \cdots y_0)$.

Phase 1 Step 2:

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \to P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_1 \overline{y_0})$

From phase 2, there are $2^{p-1}$ steps in phase $p$, where $2 \leq p \leq d-1$. In step $2i-1$, $2i$ of phase $p$, $1 \leq i \leq 2^{p-2}$, the following operations are performed.

Phase *p* Step 2*i*-1:

$If\ (iam \in G_p(2i-1))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \to P(x_{d-1} \cdots x_p \overline{x_{p-1} \cdots x_0}, y_{d-1} \cdots y_0)$

$If\ (iam \in G_p(2i))$

$$P(x_{d-1}\cdots x_0, y_{d-1}\cdots y_0) \rightarrow P(x_{d-1}\cdots x_0, y_{d-1}\cdots y_p\overline{y_{p-1}}\cdots\overline{y_0})$$

Phase *p* Step 2*i*:

$If\ (iam \in G_p(2i-1))$

$$P(x_{d-1}\cdots x_0, y_{d-1}\cdots y_0) \rightarrow P(x_{d-1}\cdots x_0, y_{d-1}\cdots y_p\overline{y_{p-1}}\cdots\overline{y_0})$$

$If\ (iam \in G_p(2i))$

$$P(x_{d-1}\cdots x_0, y_{d-1}\cdots y_0) \rightarrow P(x_{d-1}\cdots x_p\overline{x_{p-1}}\cdots\overline{x_0}, y_{d-1}\cdots y_0)$$

In phase $d$, there are $2^{d-2}$ steps, which is the same as those in phase $d-1$. In step $2i-1$ or $2i$, where $1 \le i \le 2^{d-3}$, the following operations are performed.

Phase *d* Step 2*i*-1:

$If\ (iam \in G_p(2i-1))$

$$P(x_{d-1}\cdots x_0, y_{d-1}\cdots y_0) \rightarrow P(\overline{x_{d-1}}\cdots\overline{x_0}, y_{d-1}\cdots y_0)$$

$If\ (iam \in G_p(2i))$

$$P(x_{d-1}\cdots x_0, y_{d-1}\cdots y_0) \rightarrow P(x_{d-1}\cdots x_0, \overline{y_{d-1}}\cdots\overline{y_0})$$

Phase *d* Step 2*i*:

$If\ (iam \in G_{d-1}(2i-1))$

$$P(x_{d-1}\cdots x_0, y_{d-1}\cdots y_0) \rightarrow P(x_{d-1}\cdots x_0, \overline{y_{d-1}}\cdots\overline{y_0})$$

$If\ (iam \in G_{d-1}(2i))$

$$P(x_{d-1}\cdots x_0, y_{d-1}\cdots y_0) \rightarrow P(\overline{x_{d-1}}\cdots\overline{x_0}, y_{d-1}\cdots y_0)$$

**Examples:**

Consider the following example. For a $16 \times 16$ torus, phases 1 and 2 are identical to T1. In phase 3, there are four steps. The first two steps in phase 3 are identical to T1 shown in Figures 1 (e) and (f). Steps 3 and 4 in phase 3 are shown in Figures 3 (a) and (b). In T1, nodes in $G_3(1)$ and $G_3(2)$ exchange blocks while nodes in $G_3(3)$ and $G_3(4)$ do not participate in message exchange operations in phase 3 and eventually receive data during the send phase. The key modification here is to add two more steps to phase 3 and enable these nodes in $G_3(3)$ and $G_3(4)$ to acquire blocks. This was prevented in the first two steps of phase 3 due to link contention in T1. In phase 4, there are also four steps. The first two steps in phase 4 are identical to T1 shown in Figure 3 (c), where only step 1 is shown. In step 2, active nodes in step 1 change dimensions and exchange blocks. Step 3 in phase 4 is shown in Figure 3 (d) and, in step 4, active nodes in the step change dimensions and exchange blocks. After the 4 steps in the phase 4, all processors have all blocks from all processors.

**Complexity Analysis:**

For a $2^d \times 2^d$ torus, two steps are required in phase 1. From phase 2 until phase $d-1$, there are $2^{p-1}$ steps in phase *p*, where $2 \le p \le d-1$. In phase $d$, there are $2^{d-2}$ steps. Thus, the total number of steps is $3 \cdot 2^{d-2}$. In each step, the number of exchanged blocks is $2^{2d-1}$. So, the total number of exchanged blocks is $3 \cdot 2^{3(d-1)}$. Total message start-up time is $(3 \cdot 2^{d-2})t_s$ and total message transmission time is $(3 \cdot 2^{3(d-1)})mt_c$.

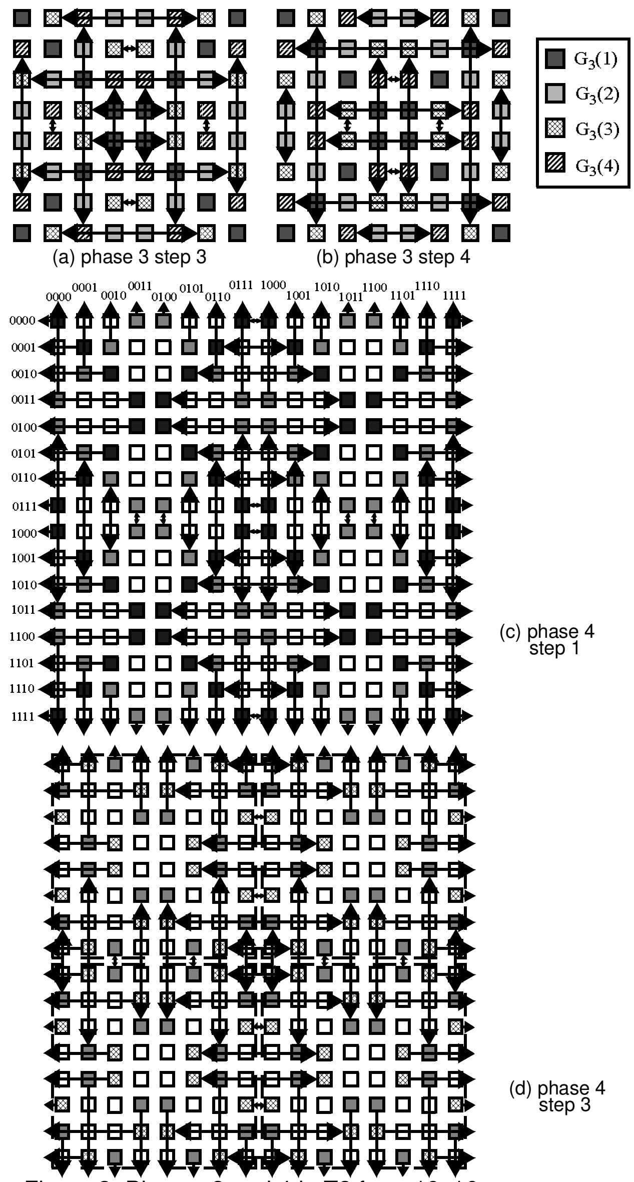



Figure 3. Phases 3 and 4 in T2 for a 16x16 torus.

# 4. Configurable Algorithms

In the previous section, we described two bottom-up algorithms: T1 and T2. Algorithm T1 focused on minimizing the message start-up cost while T2 focused on minimizing the message-transmission cost. An interesting feature of these algorithms is that they can produce a range of implementations by trading off message start-up time for message-transmission time. Within an exchange phase, a small number of additional steps may be added at the expense of a smaller send phase. Such tradeoffs can be made to balance message size against the cost of message start-ups. These algorithms improve the start-up time for T2 at the expense of message-transmission time, or vice versa. We now propose such configurable algorithms, called T1.*x*, where $x = 1, \ldots, d-4$. First, we propose algorithm T1.1, and then general algorithms T1.x are proposed.

## 4.1 Algorithm T1.1

**Communication Pattern:**

In a $2^d \times 2^d$ torus, the algorithm T1.1 consists of $d$ exchange phases followed by one send phase. In phase 1, there are two steps and the communication pattern in phase 1 is the same as the those in algorithms T1 and T2:

Phase 1 Step 1:

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \rightarrow P(x_{d-1} \cdots x_1 \overline{x_0}, y_{d-1} \cdots y_0)$

Phase 1 Step 2:

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \rightarrow P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_1 \overline{y_0})$

In phase 2, there are also two steps and nodes in $G_2(1)$ send blocks horizontally in the first step, then send blocks vertically in the second step:

Phase 2 Step 1:

$If\ (iam \in G_2(1))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \rightarrow P(x_{d-1} \cdots \overline{x_1} x_0, y_{d-1} \cdots y_0)$

Phase 2 Step 2:

$If\ (iam \in G_2(1))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \rightarrow P(x_{d-1} \cdots x_0, y_{d-1} \cdots \overline{y_1} y_0)$

In phase $p$, $3 \le p \le d-1$, there are $2^{p-1}$ Gs, among which nodes in $2^{p-2}$ Gs that constitute $G_2(1)$ participate in exchange operations in $2^{p-2}$ steps. In step $2i-1$, or $2i$, $1 \le i \le 2^{p-3}$, of phase $p$, the following operations are performed.

Phase $p$ Step $2i$-1:

$If\ (iam \in G_p(4i-3))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \rightarrow P(x_{d-1} \cdots x_p \overline{x_{p-1}} \cdots \overline{x_0}, y_{d-1} \cdots y_0)$

$If\ (iam \in G_p(4i-2))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \rightarrow P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_p \overline{y_{p-1}} \cdots \overline{y_0})$

Phase $p$ Step $2i$:

$If\ (iam \in G_p(4i-3))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \rightarrow P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_p \overline{y_{p-1}} \cdots \overline{y_0})$

$If\ (iam \in G_p(4i-2))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \rightarrow P(x_{d-1} \cdots x_p \overline{x_{p-1}} \cdots \overline{x_0}, y_{d-1} \cdots y_0)$

In phase $d$, nodes in $2^{d-3}$ Gs that constitute $G_2(1)$ participate in exchange operations in $2^{d-3}$ steps, where message exchanges among one half of nodes are performed using local channels while those among the other half of nodes are performed using wrap-around channels. In step $2i-1$ or $2i$, $1 \le i \le 2^{d-4}$, of phase $d$, the following operations are performed.

Phase $d$ Step $2i$-1:

$If\ (iam \in G_p(4i-3))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \rightarrow P(\overline{x_{d-1}} \cdots \overline{x_0}, y_{d-1} \cdots y_0)$

$If\ (iam \in G_p(4i-2))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \rightarrow P(x_{d-1} \cdots x_0, \overline{y_{d-1}} \cdots \overline{y_0})$

Phase $d$ Step $2i$:

$If\ (iam \in G_p(4i-3))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \rightarrow P(x_{d-1} \cdots x_0, \overline{y_{d-1}} \cdots \overline{y_0})$

$If\ (iam \in G_p(4i-2))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \rightarrow P(\overline{x_{d-1}} \cdots \overline{x_0}, y_{d-1} \cdots y_0)$

After phase $d$, each node in $G_2(1)$ has all blocks from all other nodes, destined for itself and a neighboring node. In one step in the send phase, each node in $G_2(2)$ receives blocks destined for itself from a node in $G_2(1)$ as follows.

Send Phase Step 1:

$If\ (iam \in G_2(1))$

$P(x_{d-1} \cdots x_0, y_{d-1} \cdots y_0) \rightarrow P(x_{d-1} \cdots x_1 \overline{x_0}, y_{d-1} \cdots y_0)$

**Example:**

Consider an example for a $32 \times 32$ torus. The communication patterns until phase 3 are the same as those of T1 shown in Figures 1 (a)-(f). The communication operations in phase 4 are illustrated in Figure 4. Due to channel contention, nodes in $G_2(1)$ cannot exchange blocks in two steps. So, in the first two steps, nodes in $G_4(1)$ and $G_4(2)$ (i.e., nodes in $G_3(1)$) exchange blocks, then nodes in $G_4(5)$ and nodes in $G_4(6)$ (i.e., nodes in $G_3(2)$) exchange blocks in the next two steps. Figure 5 shows the communication patterns in phase 5. In the first two steps of phase 5, nodes in $G_4(1)$ and $G_4(2)$ exchange blocks, then nodes in $G_4(5)$ and nodes in $G_4(6)$ exchange blocks in the next two steps. Now, each node in $G_2(1)$ possesses all blocks from all nodes in the $32 \times 32$ torus destined for itself and a neighbor node with which it exchanged blocks in step 1 of phase 1. Thus, in a single step in the send phase, the blocks destined for the neighbor node are transmitted as shown in Figure 2(c).

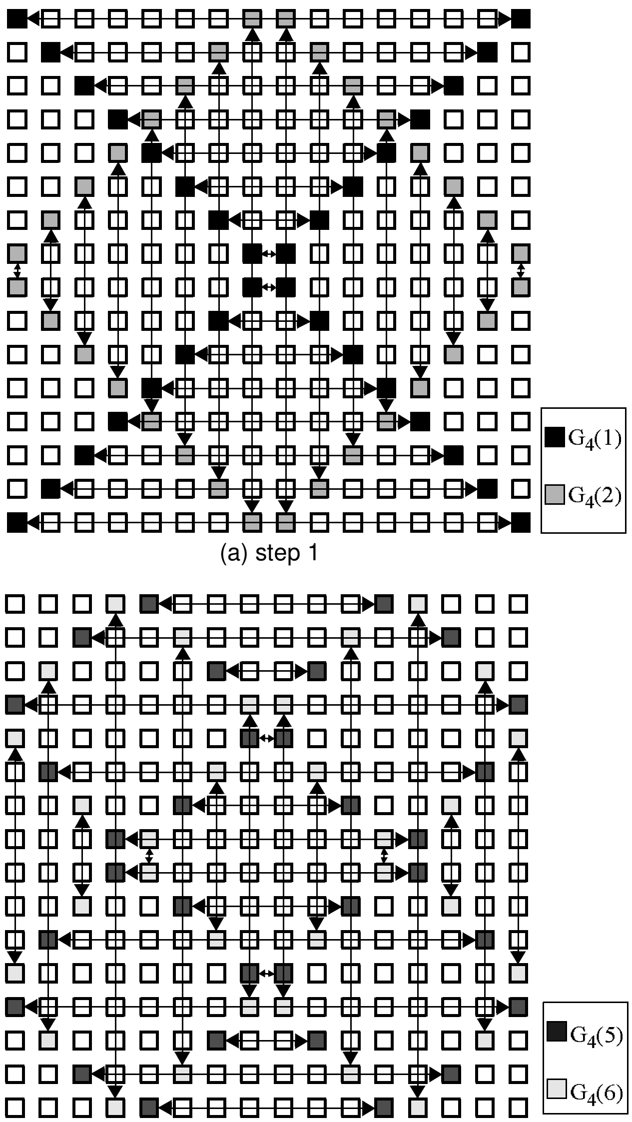
**Complexity Analysis:**

For a $2^d \times 2^d$ torus, there are two steps per phase in phases 1 and 2. In phase $p$, $3 \le p \le d-1$, there are $2^{p-2}$ steps. In phase $d$, there are $2^{d-3}$ steps and there is one step in the send phase. Thus, the total number of steps is $3 \cdot 2^{d-3} + 3$. In the first step of phase 1, $2^{2d-1}$ blocks are transmitted. In each step of the remaining $3 \cdot 2^{d-3} + 2$ steps, the number of transmitted blocks is $2^{2d}$. Thus, the total number of transmitted blocks is $3 \cdot 2^{3d-3} + 5 \cdot 2^{2d-1}$.

## 4.2 Algorithm T1.x

**Communication Pattern:**

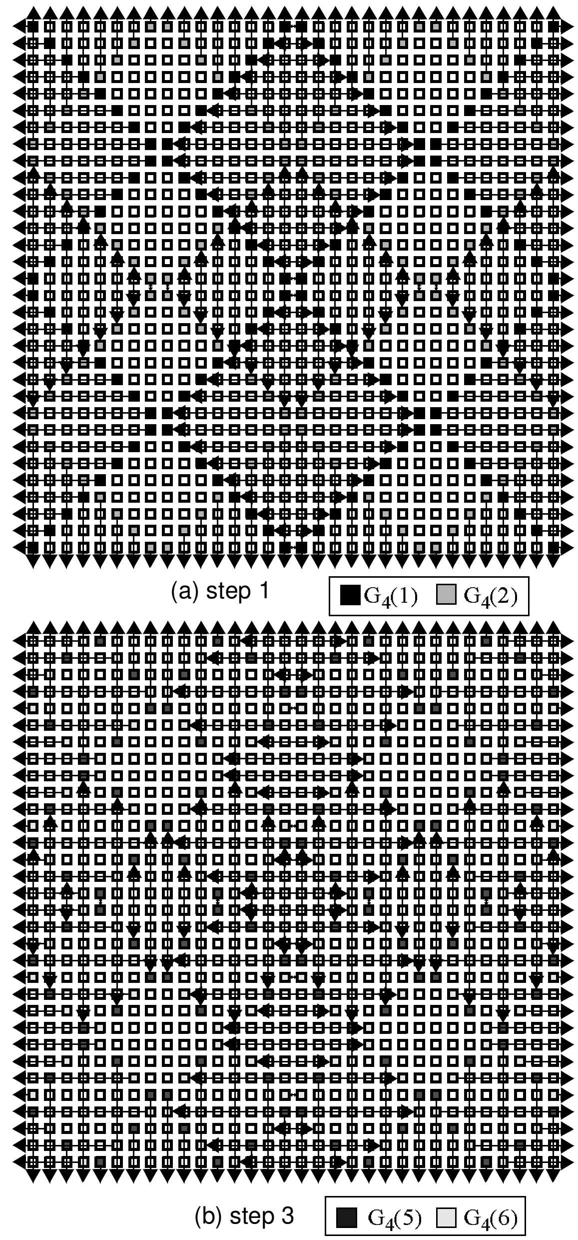
Algorithm T1.$x$ is defined for a $2^d \times 2^d$ torus, where $d \ge x+4$. In a $2^d \times 2^d$ torus, T1.$x$ consists of $d$ exchange phases followed by one send phase which consists of $x$ steps. Until phase $x+1$, there are two steps per phase and communication patterns are very similar to those in Algorithm T1. After phase $x+1$, nodes in $G_{x+1}(1)$ have all blocks from nodes in a $2^{x+1} \times 2^{x+1}$ submesh. From phase $x+2$, nodes in $G_{x+1}(1)$ cannot exchange blocks in two steps due to channel contention. In phase $p$,



(a) step 1

(b) step 3

Figure 4. Phase 4 (in each 16x16 submesh) in algorithm T1.1 for a 32x32 torus.



(a) step 1

(b) step 3

Figure 5. Phase 5 in algorithm T1.1 for a 32x32 torus.

$x+2 \le p \le d-1$, there are $2^{p-1}$ Gs, among which nodes in $2^{p-2}$ Gs that constitute $G_{x+1}(1)$ participate in exchange operations in $2^{p-2}$ steps. In phase $d$, there are $2^{d-3}$ steps. In $x$ steps of the send phase, each node not in $G_{x+1}(1)$ receives blocks destined for itself from a node in $G_{x+1}(1)$.

**Complexity Analysis:**

For a $2^d \times 2^d$ torus, $d \ge x+4$, there are two steps per phase until phase $x+1$. In phase $p$, $x+2 \le p \le d-1$, there are $2^{p-2}$ steps. In phase $d$, there are $2^{d-3}$ steps and there are $x$ steps in the send phase. Thus, the total number of steps is $(3 \cdot 2^{d-3}+3x+2)-2^x$. In the first $2x-1$ steps, $2^{2d+x-2}$ blocks are transmitted in each step. In the remaining $(3 \cdot 2^{d-3}+3)-2^x$ steps in exchange phases, $2^{2d+x-1}$ blocks are transmitted in each step. In step $s$ of the send phase, where $1 \le s \le x$, $2^{2d+x-s}$ blocks are transmitted. Thus, the total number of transmitted blocks is $3 \cdot 2^{3d+x-4}+(2x+3)2^{2d+x-2}$.

## 5. Performance Evaluation

Here, we have presented a set of configurable complete exchange algorithms for 2D tori, which operate in a bottom-up fashion proceeding from contiguous $2 \times 2$ submeshes to higher-order submeshes.

In Algorithm T1, exchanges continue in larger and larger submeshes, until some of the processors in each quadrant have blocks from all of the processors. Getting to this point requires a relatively few steps since all processors in each quadrant need not have the blocks from all processors. At this point a send phase can be initiated. The basic idea of Algorithm T1 is that the number of steps required for some processors to acquire all the blocks is relatively small. The additional number of steps in the send phase also grows slowly. The small number of steps, in turn, reduces the total start-up time. T1 focused on minimizing start-up cost at some expense of message-transmission cost, whereas T2 focused on minimizing message-transmission cost with an increased start-up cost. Existing algorithm [19] has identical characteristic as T2 and thus the performance of T2 is very similar to that of the algorithm [19]. When software overhead of message initiation is high, T1 is preferred, and when message-transmission time is dominant, T2 is preferred. The variants between the two are possible where the time for message start-ups can be traded against larger message sizes. This is useful in configuring the algorithm for different message-passing machines based on message-initiation overhead and link speeds. A set of configurable algorithms of T1.$x$ has been developed for this purpose.

**Table 1: Performance summary of algorithms.**

| | Range of d | Start-up Time | Message Transmission Time |
|---|---|---|---|
| T1 | $d < 4$ | $(2d)t_s$ | $(d \cdot 2^{2d})m \cdot t_c$ |
| | $d \geq 4$ | $(3d-3)t_s$ | $\{9 \cdot 2^{3d-4} + (d^2 - 5d + 3)2^{2d-1}\} m \cdot t_c$ |
| T2 | $d < 4$ | $(2d)t_s$ | $(d \cdot 2^{2d})m \cdot t_c$ |
| | $d \geq 4$ | $(3 \cdot 2^{d-2})t_s$ | $(3 \cdot 2^{3d-3})m \cdot t_c$ |
| T1.x | $d > x+4$ | $\{3 \cdot 2^{d-3} + 3x + 2 - 2^x\} t_s$ | $\{3 \cdot 2^{3d+x-4} + (2x+3)2^{2d+x-2}\} m t_c$ |

The time complexities of T1, T2, and T1.x are summarized in Table 1. For $2^d \times 2^d$ tori, message start-up costs are $O(2^d)$ for T2 while they are $O(d)$ for T1. Algorithms T1.$x$ also show $O(2^d)$ start-up cost and $O(2^{3d})$ message-transmission cost. But, they have start-up costs that are higher than T1 but lower than T2. In addition, message-transmission costs are lower than T1 and higher than T2.

Ideally, we would like to evaluate the performance of these algorithms on commercial parallel supercomputers. However, evaluation of their scalability across a range of system sizes is hampered by the unavailability of large systems, and by the lack of control over the shape of submeshes allocated in commercial sizeable machines. What we need is a more flexible methodology that would yield reliable estimates of execution time across a broader range of system sizes. Consequently, we use analytic models of execution time that are based on real values of parameters measured on commercial machines. Where relevant, the values of parameters were measured as a function of problem size. In order to derive the interconnection network parameters, roundtrip test messages of known (large) size were transmitted between pairs of processors. The measured times were averaged over the number of messages. Given this parameterized model, with values of $t_c$ and $t_s$, it is now possible to study the performance of different

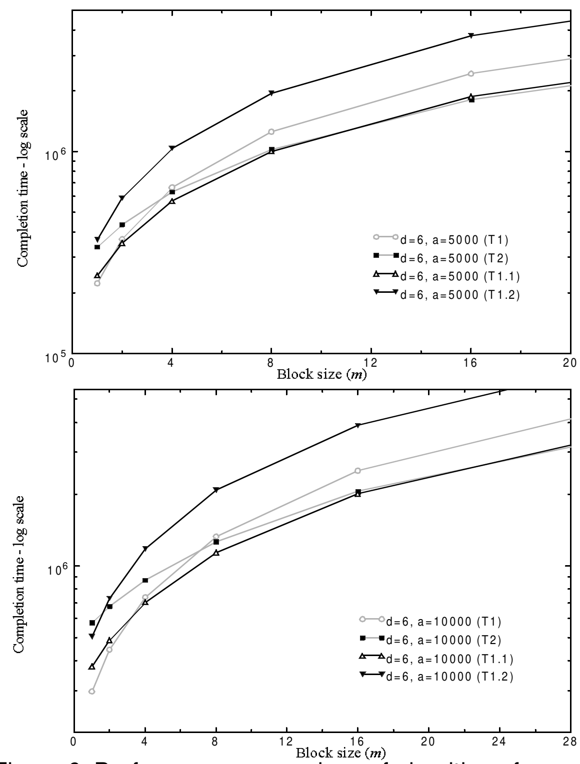



Figure 6. Performance comparison of algorithms for a $2^6$x$2^6$ torus.

algorithms over a wide range of systems and problem sizes without requiring access to the machine configurations of these sizes. While not as realistic as time measured on real implementations, the model is detailed enough to provide insight into the performance over a range of system parameters.

Figures 6 and 7 illustrate the completion times of T1, T2, and T1.x for different network sizes as a function of block size, while considering only the two dominant terms: start-up cost and message-transmission cost. In this analysis, the comparison is parameterized with the normalized ratio of $t_s$ to $t_c$, $a$ assuming that $t_c$ has a value of one time unit. According to our measurements of the start-up costs and message transmission costs in commercial multicomputers such as Intel Paragon or Cray T3D, we found that the normalized ratio $a$ is between 2500 to 10000. So, we evaluate the performance of the proposed algorithms for two values of $a$ in this range, 5000 and 10000. Note that, since existing algorithms and the proposed algorithm T2 yield almost the same performance [13], we compare the performance of T1, T2, and T1.x.

Figure 6 shows the completion times of T1, T2, T1.1, and T1.2 in a $64 \times 64$ torus as functions of block size and the ratio $a$. The completion time plot indicates that when the block size is small, T1 is shown to have the best performance while T2 or T1.2 shows the worst performance. As the block size increases, T1.1 exhibits the best performance, while T1 becomes worse. When the block size further increases, T2 exhibits the best performance. Note that the cross-over points both between T1 and T1.1, and T1.1 and T2 appear in large block sizes as the ratio $a$ increases. It indicates that as start-up cost is a more dominant factor,

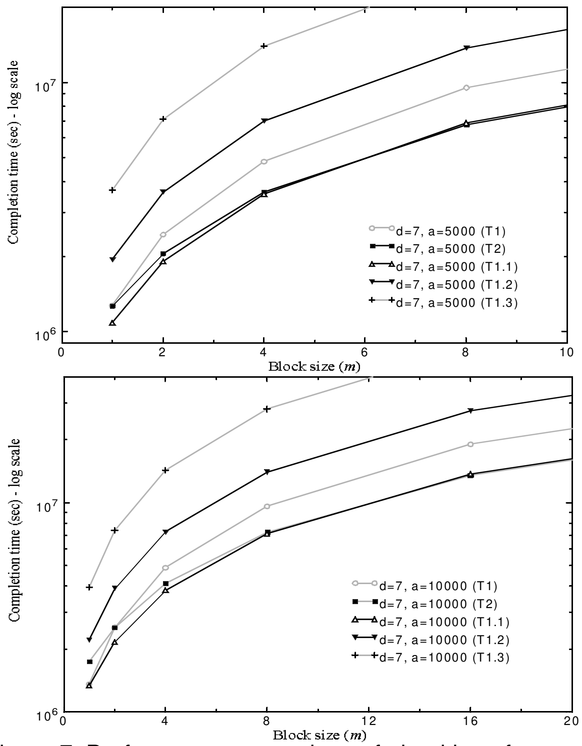



Figure 7. Performance comparison of algorithms for a $2^7 \times 2^7$ torus.

T1 provides the best performance until larger block sizes, while T2 shows the best performance for larger block sizes. Between ranges of the two points of block sizes, T1.1 shows the best performance. In this network, T1.2 does not show good performance. Figure 7 shows the completion times of T1, T2, T1.1, T1.2, and T1.3 in a $128 \times 128$ torus as functions of block size and the ratio $a$. The general characteristics of these plots are very similar to those in Figure 6. But, the cross-over points appear in smaller block sizes, indicating that, as the network size increases, T1 becomes a less powerful algorithm while T2 becomes more efficient. In this network, T1.2 and T1.3 do not show good performance.

From the above performance figures, we can observe some important facts: i) in a given network, T1, T1.1, T2 show the best performance when block sizes are relatively small, medium, and large, respectively, ii) in a given network, T1 shows better performance when the start-up cost is dominant, iii) for a given performance parameters, T2 shows better performance in larger networks and larger block size.

## 6. Conclusion

In this paper, we presented a set of configurable algorithms for complete exchange for two-dimensional torus-connected networks. They can be tuned to trade message-initiation or start-ups overhead against message-transmission time. The ability to configure these algorithms allows us to match the algorithm characteristics with machine characteristics based on message-initiation overhead and link speeds, in order to minimize overall execution time.Our performance evaluation results have shown T1, T1.1, T2 to perform best when block sizes are relatively small, medium, and large, respectively in a given network.

## References

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

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

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

[4] W. J. Dally, "Performance Analysis of *k*-ary *n*-cube Interconnection Networks," *IEEE Trans. on Computer*, vol. 39, no. 6, pp. 775-785, June 1992.

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

[6] 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.

[7] P. K. McKinley, Y.-J. Tsai, and D. F. Robinson, "Collective Communication Trees in Wormhole-Routed Massively Parallel Computers," Technical Report MSU-CPS-95-6, Michigan State University, March 1995.

[8] P. K. McKinley and Y.-J. Tsai and D. Robinson, "Collective Communication in Wormhole-routed Massively Parallel Computers," IEEE Computer, pages 39--50, December 1995.

[9] L. M. Ni and P. K. McKinley, "A Survey of Wormhole Routing Techniques in Direct Networks," *IEEE Computer*, vol. 26, pp. 62-76, February 1993.

[10] D. K. Panda, "Issues in Designing Efficient and Practical Algorithms for Collective Communication on Wormhole-Routed Systems," Technical Report TR-25, Dept. of Computer and Information Science, Ohio State University.

[11] 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.

[12] Y. J. Suh and S. Yalamanchili, "Algorithms for All-to-All Personalized Exchange in 2D and 3D Tori," Proceedings of the 10th International Parallel Processing Symposium, pages 808-814, April 1996.

[13] Y. J. Suh and S. Yalamanchili, "All-to-All Communication with Minimum Start-Up Costs in 2D/3D Tori and Meshes," IEEE Transactions on Parallel and Distributed Systems, Vol. 9, No. 5, May 1998.

[14] Y. J. Suh and K. G. Shin, "Efficient All-to-All Personalized Exchange in Multidimensional Torus Networks," Proceedings of the 27th International Conference on Parallel Processing, August 1998.

[15] Y. J. Suh, K. G. Shin, and S. Yalamanchili, "Complete Exchange in General Multidimensional Mesh Networks," Proceedings of the 10th International Conference on Parallel and Distributed Computing Systems, 1997.

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

[17] 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.

[18] 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

[19] 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.
[20] Message Passing Interface Forum, "MPI: A Message-Passing Interface Standard," Technical Report CS-93-214, University of Tennessee, April 1994.
[21] Cray T3D, *System Architecture Overview*, 1994.