Fault-Tolerant Broadcasting Algorithms in Two Dimensional Circuit-Switched Torus Network

N.W. Lo, Bradley S. Carlson and *D.L. Tao*

The Department of Electrical Engineering
State University of New York at Stony Brook
Stony Brook, NY 11794-2350
Fax: (516)632-8494
Phone: (516)632-8474
Internet: bcarlson@jessica.ee.sunysb.edu

Technical Report: #745

Abstract

In this paper optimal fault tolerant broadcasting algorithms for 2-dimensional synchronous circuit-switched torus network are presented. In the presence of up to three faults in a 2−D torus network the proposed algorithms can perform one-to-all broadcasting to all fault-free processors, provided that the minimum number of broadcasting steps are used.

Keywords: circuit-switched broadcast, 2-D torus, fault tolerance

1 Introduction

New transmission protocols, enhanced communication technology, and new transmission medium together provide multicomputer systems a very good candidate to achieve cost-effective high-performance computing tasks. Multicomputer systems are composed by a set of microprocessor systems in which each node has its own local memory and computing power. Point-to-point network is used to connect nodes and message-passing mechanism is implemented to exchange information between nodes. The performance of message-passing distributed memory multicomputer systems is dependent on the network topology and the routing mechanism.

A broadcasting problem is to distribute original information from a source node to all other nodes under a fixed network topology and specific routing mechanism. Among several different network topologies [?] and routing schemes [?], a 2-dimension torus network along with circuit-switched routing mechanism appears to be a very popular and efficient combination for today's multicomputer systems [?, ?]. The low degrees and symmetric geometry characteristics of 2-dimension torus make this topology easy to construct under any interconnection networks. On the other hand the circuit-switched routing provides a contention-free distance-insensitive routing mechanism [?, ?]. The analysis shown in [?] indicates the general transmission times for the two different circuit-switched routing types, direct-connect routing and wormhole routing, are similar.

Recently the adoption of parallel algorithms onto existent multicomputer systems to achieve high-performance computation is a popular research area. In order to preserve the computation power and the correctness of computing results, the fault tolerant capability of nodes of a multicomputer system is an important factor [?, ?].

In this paper new fault tolerant broadcasting algorithms based on Peters and Syska's [?] optimum broadcasting algorithm are developed for a $5^p \times 5^p$ ($p \in Z^+$) 2-dimension synchronous circuit-switched torus network. In this paper the direct-connect routing model is adopted for its simplicity and synchronous characteristics. In the presence of up to three faults in a $2-$D torus network the proposed algorithms can perform one-to-all broadcasting to all fault-free processors, provided that the minimum number of broadcasting steps are used.

The remainder of this paper is presented as follows. The communication model is described in Section ??. The optimal broadcasting algorithm derived by Peters and Syska [?] is reviewed in Section ??. Our fault tolerant broadcasting algorithms and relative implementation issues are described in Section ?? and the paper is concluded in Section ??.

2 Communication Model

A $5^p \times 5^p$ 2-dimension torus network $T = (V, E)$ where V represents the nodes in the torus and E contains all corresponding link edges between nodes is defined as $V = \{(i,j) \mid 0 \leq i,j \leq 5^p - 1, p \in Z^+\}$ and $E = \{((u_1,v_1),(u_2,v_2)) \mid u_1 = (u_2 \pm 1) \bmod 5^p$ and $v_1 = v_2$ OR $u_1 = u_2$ and $v_1 = (v_2 \pm 1) \bmod 5^p \ \forall (u_1,v_1),(u_2,v_2) \in V, p \in Z^+\}$. We assume the node $(0,0)$ is in the left-bottom corner of a torus graph and the other three corner nodes are labeled $(0, 5^p - 1), (5^p - 1, 5^p - 1)$, and $(5^p - 1, 0)$, respectively. The assumptions for network characteristics are the following:

- *Link-bounded* model: a node (processor) can read and write through all its communication links simultaneously.
- Communication link is in *full-duplex* mode, i.e., both directions can be used to transfer data at the same time.
- A node can pass an incoming message from one of its input ports to one of its output ports. Up to four messages can be switched through a node at the same time when no two incoming messages are directed to the same output port.
- *Circuit-switched routing (direct-connect)*: before transferring any message a connection request header must be sent from the source node and received by the destination node to establish the communication path between the source node and destination node. After receiving a connection request the destination node sends an acknowledgement header back to the source node to confirm the path is established. By adopting the direct-connect routing type we avoid the contention and deadlock problems in wormhole routing mechanism.

We only consider permanent node failures which also indicates all four corresponding links of a faulty node are inoperable. Under the assumption that only one fault occurs at any given time we can broadcast the address of the faulty node from one of its alive neighbor nodes to all other fault-free nodes in the network by applying our fault tolerant broadcast algorithms. Therefore, we can assume the addresses of faulty nodes are known by every non-faulty node. In other words, every healthy node contains global information about the locations of faulty nodes in the torus network.

In a stable interconnection network environment the number of faulty nodes (processors) in a reasonable period of time should not be more than 1% of the total number of nodes. Our 3–fault tolerant algorithms presented in Section ?? is mainly suitable for the multicomputer with its total number of nodes up to several hundreds.

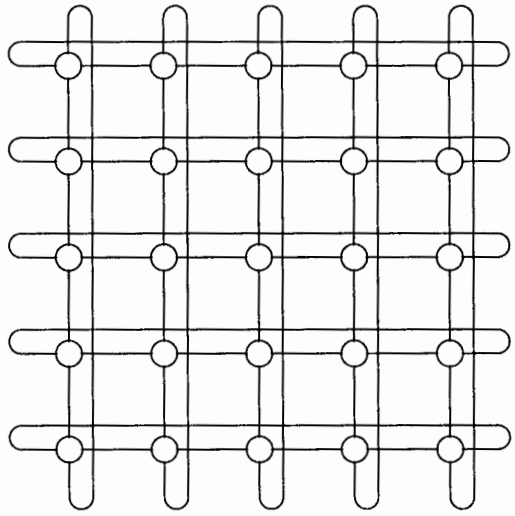


Figure 1: The 5 x 5 torus graph.

3 Optimal Broadcasting Algorithm for 2-D Torus

In [?] Peters and Syska presented a 2-dimensional torus broadcasting algorithm in a fault free network and proved their algorithm is optimal from their theorem 2. For the sake of simplicity Peters and Syska's broadcasting algorithm is denoted as the *PSB algorithm* from now on. The *PSB algorithm* can be viewed as a recursive tiling of a torus. To illustrate their algorithm we consider the example of a 5 x 5 2-dimensional torus shown in Fig. ?? and assume the node located at the left-bottom corner is labeled $(0,0)$. Fig. ?? shows two phases of the *PSB algorithm* in a 5 x 5 2-dimensional torus. The black node in the center of the graph is assumed to be the source node with label $(2,2)$. In the first phase the source node broadcasts the message to the four black nodes with labels $(4,1),(1,0),(0,3)$, and $(3,4)$ via four disjoint paths marked with large arrows. In the second phase each black node (u,v), including the source node $(2,2)$, sends the message to its four immediate neighbors $(u+1,v),(u-1,v),(u,v+1)$, and $(u,v-1)$. If we unwrap the 5 x 5 torus carefully, we can get a mesh-form diagram shown in Fig. ??.

Definition 1: A *cross unit* is defined as one node and its four neighbor nodes connected together with its four links. A *basic group unit*, a mesh-form diagram of 5 x 5 torus, is composed of 5 cross units. A *level-2 cross unit* contains five basic group units that one basic group unit is positioned at the center with the four other basic group units connected along each edge.

By adopting the recursive tiling scheme into the *PSB algorithm* a broadcast can be completed in $2p$ broadcasting steps from an originator (source node) in any 2-dimensional torus graph of size $N = 5^p \times 5^p$. Fig. ?? shows a 25 x 25 torus drawn as a mesh form which is composed of five level-2 cross units (subgraphs). Some of the details of the *PSB algorithm* are omitted from the four outer

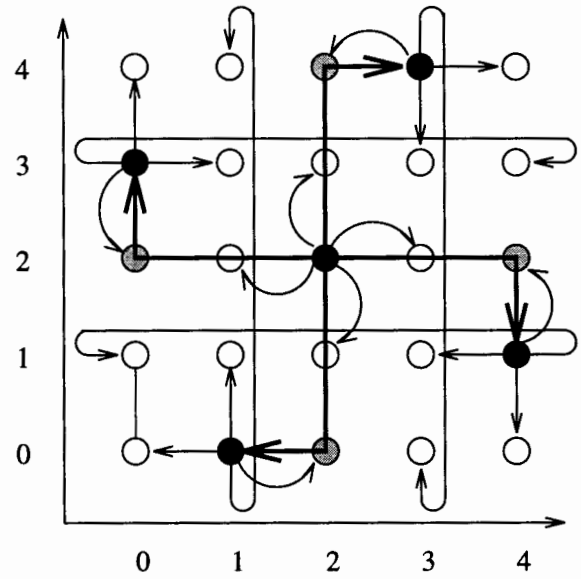


Figure 2: Broadcasting in a 5×5 torus graph.

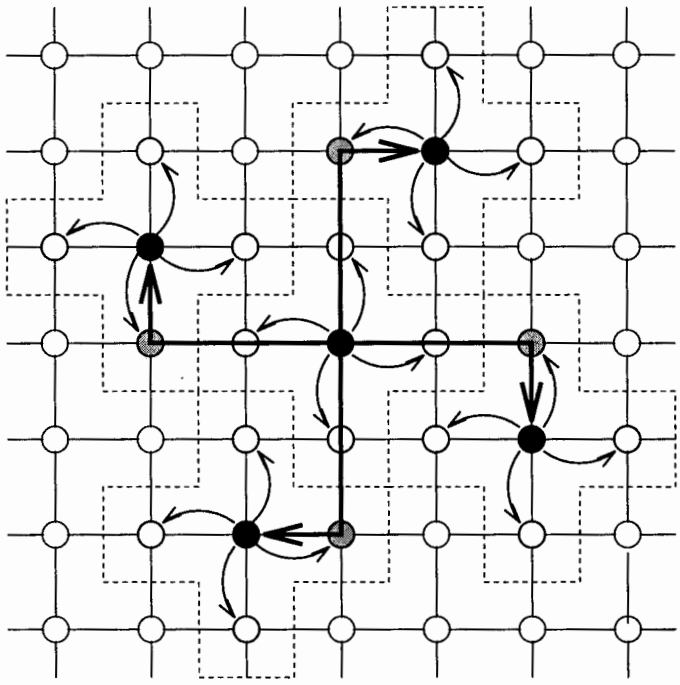


Figure 3: A broadcast process on a 5×5 torus which is composed of 5 cross units drawn as a mesh form.

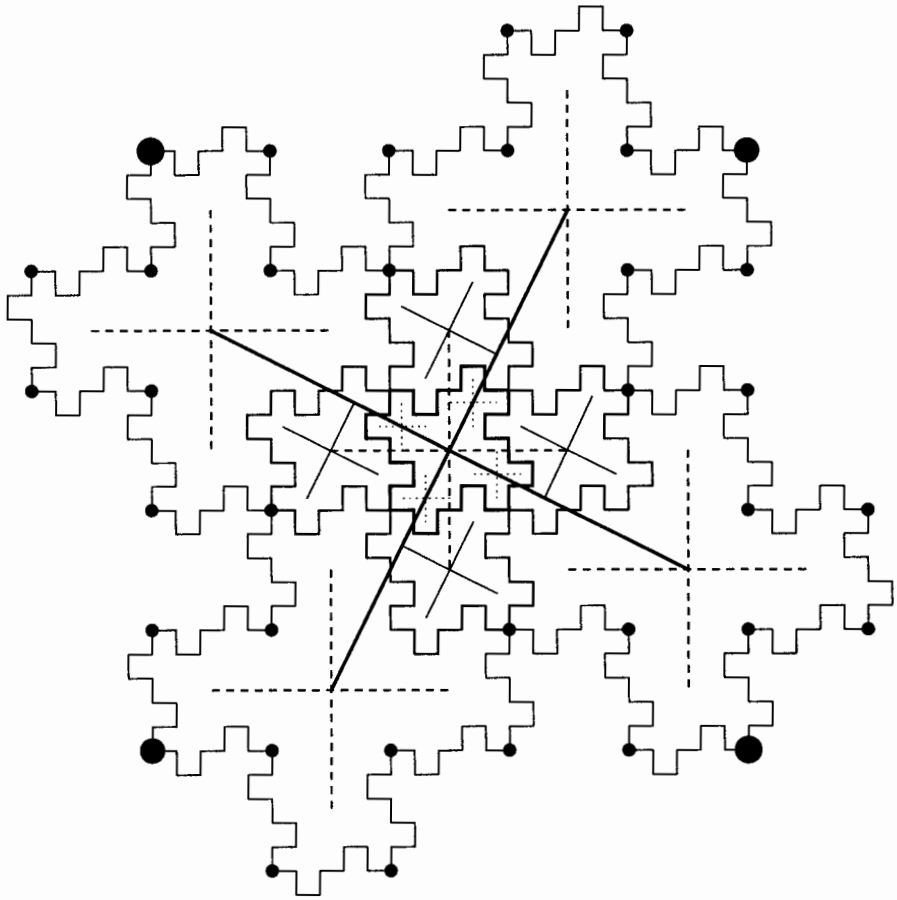


Figure 4: A broadcast process on a 25 × 25 torus drawn as a mesh form.

level-2 cross units to simplify the diagram. The recursive tiling scheme can be observed in this example. During a broadcast process the center source node in the center subgraph first sends the message to each center node of the other four subgraphs. Each subgraph is composed of 5 basic group units. Then at the second step every center node of these 5 subgraphs sends the message to the center of the other four basic group units in its own subgraph simultaneously. Before the third broadcasting step, each basic group unit in the 25 × 25 torus has the message in its center node. After two more steps are applied in each basic group unit (as shown in Fig. ??) the broadcast task is complete.

4 Fault Tolerant Algorithms for 2-D Torus

Without loss of generality we assume the originator is located at the center of the $5^p \times 5^p$ torus network with label $(x, y) = (\frac{5^p-1}{2}, \frac{5^p-1}{2})$ on itself and the left-bottom corner node is labeled $(0, 0)$. For consistency we adopt the same notation to represent a 2-D torus in the rest of this paper.

Definition 2: In a basic group unit the *root node* is the center node of the center cross unit. A

branch node is one of the four center nodes of surrounding cross units. A *pre-branch node* is one of the four nodes in surrounding cross units with its X-axis or Y-axis position labels equal to the ones of the root node. The rest of nodes in a basic group unit are defined as *leaf nodes.*

As shown in Fig. ?? and Fig. ?? in a basic group unit the root node is the black node at the center of these diagrams. Branch nodes are black nodes around the root node. Pre-branch nodes are painted in gray color. All white nodes denote leaf nodes.

Definition 3: An embedded torus $G \subset T$ is defined as $G = (V, E)$ such that

1. The broadcasting center node is labeled $(a, b) = (\frac{5^p-1}{2}, \frac{5^p-1}{2})$.
2. $V = \{(i, j) \mid\mid i - a \mid = 5m, \mid j - b \mid = 5n, 0 \leq m, n \leq \lfloor \frac{5^p-1}{10} \rfloor, p \geq 2\}$.
3. $E = \{((u_1, v_1), (u_2, v_2)) \mid u_1 = (u_2 \pm 5) \bmod 5^p$ and $v_1 = v_2$ OR $u_1 = u_2$ and $v_1 = (v_2 \pm 5) \bmod 5^p \; \forall (u_1, v_1), (u_2, v_2) \in V, p \geq 2\}$.

Definition 4: A node set, $L(V) \subset T(V)$, is defined as the nodes in $T(V)$ located on any link edge in G.

The idea to tolerate up to three faulty nodes in a $5^p \times 5^p$ 2-dimensional torus network T is to separate the nodes of the torus graph into two subsets, $G(V)$ and $T(V) - G(V)$. The subset $G(V)$ contains all nodes that will receive the message during the first $2(p-1)$ broadcasting steps while no node is failed and the *PSB algorithm* is applied. Notice that all nodes in $G(V)$ form a $5^{p-1} \times 5^{p-1}$ 2-dimensional torus $G(V, E)$ into torus T. We apply a new algorithm called *find_pseudo_source_node algorithm* to find a fault-free pseudo-source node to substitute the originator if either one or both of these two cases listed in the following encounter:

- One of the faulty nodes is a branch node of a basic group unit where the root node of this basic group unit is in $G(V)$.
- One of the faulty nodes is located in $G(V) \cup L(V)$.

After applying the *find_pseudo_source_node algorithm* all the first $2(p-1)$ stages of the *PSB algorithm* are able to send the message successfully without any transmission failure because of a faulty node on the transmission path and in the worst case only one faulty node is a branch node of some basic group unit in which the root node of this basic group unit is in the new $G(V)$ constructed from the pseudo-source node. In other words, the new source node spans a new embedded torus G' such that all nodes in $G'(V) \cup L'(V)$ are fault-free.

Because there are at most three faulty nodes under our assumption and each node can find four disjoint paths to communicate with another node in the torus network, we can always find at least one fault-free connected path from the originator to the non-faulty pseudo-source node.

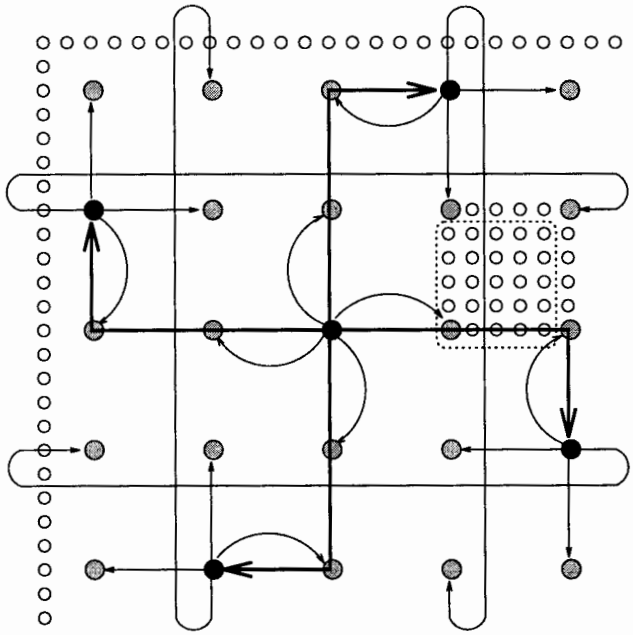


Figure 5: Broadcasting in a 25 x 25 torus graph.

A simple greedy algorithm similar to the *dimensional routing* scheme can be applied to perform the transmission task as follows. For simplicity, assume the message is sent from the orginator through its X-axis output port. The algorithm transmits the message along the X-axis direction until the Y-axis coordinates of the destination node and an intermediate node are matched. Then the message is transmitted from the Y-axis output port of the intermediate node along the Y-axis direction to its destination (the pseudo-source node). Since the originator knows the locations of the faulty nodes in advance, during the connection request the algorithm can bypass the faulty nodes in front of the connection path by changing the path direction before encountering any faulty node and switch back to the original direction later.

In Fig. **??** we show an embedded 5 x 5 torus G (all black and gray nodes) in a $5^2 \times 5^2$ torus network T. Assume node $(u, v) \in T(V) - G(V)$. From observation, node (u, v) must fall in a 5 x 5 square grid partitioned by the embedded torus G. For example, the dotted square region of Fig. **??** shows a 5 x 5 square grid constructed by a 5 x 5 embedded torus in a $5^2 \times 5^2$ torus network. Consider a 5 x 5 square grid shown in Fig. **??**. If we assume the originator labeled (x, y) is located at the left-bottom corner of the square grid in Fig. **??**, then the 4 gray nodes with labels $(x+1, y+2), (x+2, y+4), (x+3, y+1)$, and $(x+4, y+3)$ are indicated as the corresponding branch nodes of the 4 black nodes in $G(V)$. Because a torus network is totally symmetric, the whole torus network T can be partitioned into multiple identical 5 x 5 square grids as the one shown in Fig. **??**. We define a 5 x 5 virtual square grid as *the mapping square grid.* All square grids partitioned in

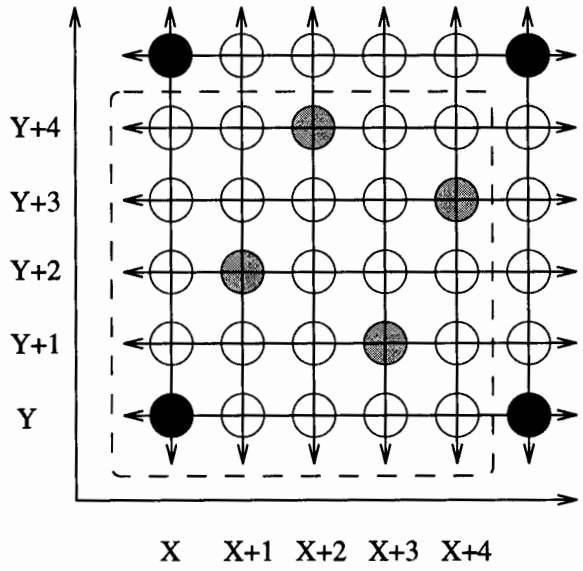


Figure 6: The 4 gray nodes represent the branch nodes of the 4 black nodes located in $G(V)$; the range of a 5 × 5 square grid is indicated by dashed lines.

a 2-D torus can superimpose to the mapping square grid. If we select the root node and the four branch nodes in a square grid, when we move around the embedded torus we can observe that these five nodes fall into different relative positions with respect to their new corresponding square grids. Mapping all different square grid partition patterns of these five nodes into the mapping square grid, it is observed that the elements (nodes) in the mapping square grid can be partitioned into 5 groups with 5 elements in each of them as shown in Fig. **??**. In general, there are 5 × 5 different relative positions for an embedded torus G to move to and become a new embedded torus G'. That is, a total of 25 different embedded torus networks can be formed in any $5^p \times 5^p$ torus network. Notice that a 5 × 5 torus ($p = 1$) itself is a square grid. Therefore, the *find_pseudo_source_node algorithm* can be applied to a 5 × 5 torus as well. We denote two link edges passed through the left and bottom sides of a square grid in G as the left and bottom boundaries of a square grid, respectively. We apply the characteristics of the square grid to define the left and bottom boundary offsets of a node (u, v) as follows.

Definition 5: A *left boundary offset* of a node (u, v), $LB(u, v)$, is the distance between the node (u, v) and the left boundary of the 5 × 5 square grid which contains the node (u, v).

Definition 6: A *bottom boundary offset* of a node (u, v), $BB(u, v)$, is the distance between the node (u, v) and the bottom boundary of the 5 × 5 square grid which contains the node (u, v).

Assume the original source node is labeled (x, y). To compute boundary offsets of a node a simple algorithm is derived as follows.

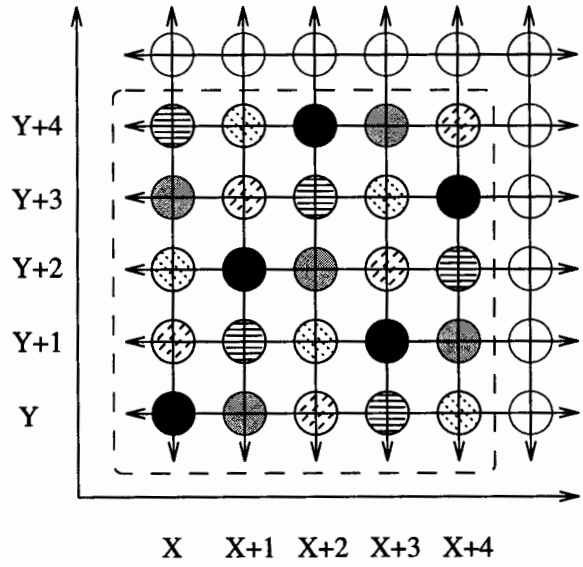


Figure 7: Five different groups in the mapping square grid.

- A left boundary offset of a node (u, v) can be computed as $(u-x) \bmod 5$ if $u-x \geq 0$; otherwise, the value is equal to $5 - (x - u) \bmod 5$.
- A bottom boundary offset of a node (u, v) can be computed as $(v - y) \bmod 5$ if $v - y \geq 0$; otherwise, the value is equal to $5 - (y - v) \bmod 5$.

Consider three possible faulty nodes (fx_1, fy_1), (fx_2, fy_2), and (fx_3, fy_3). Define the corresponding left and bottom boundary offsets of the three faulty nodes as $LB(fx_1, fy_1)$, $BB(fx_1, fy_1)$, $LB(fx_2, fy_2)$, $BB(fx_2, fy_2)$, $LB(fx_3, fy_3)$, and $BB(fx_3, fy_3)$. Let the set $F = \{(fx_1, fy_1)$, (fx_2, fy_2), and $(fx_3, fy_3)\}$. Let the set $R = \{(LB(u, v), BB(u, v)) \mid (u, v) \in$ the branch nodes of the left-bottom root node in the mapping square grid $\} = \{(1, 2), (2, 4), (3, 1), (4, 3)\}$ represent the relative positions between the branch nodes and the left-bottom node in a square grid, i.e., the four gray nodes shown in Fig. **??**. Define the final X and Y-axis offsets between the originator and the selected pseudo-source node as x_{offset} and y_{offset} where $0 \leq x_{offset}, y_{offset} \leq 4$. In order to trace the relative positions between the faulty nodes and the left-bottom node of their new corresponding square grids in accord with a possible pseudo-source node $(x + x_{offset}, y + y_{offset})$, we define a relative distance operator, $RDO(a, b)$ in which $0 \leq a, b \leq 4$, such that if $a - b < 0$, then $RDO(a, b) = a - b + 5$; otherwise, $RDO(a, b) = a - b$. Notice that in the *find_pseudo_source_node algorithm* the operands of RDO (relative distance operator) can be a pair of $LB(u, v)$ and x_{offset} variable to reference a new left boundary offset or a pair of $BB(u, v)$ and y_{offset} variable to reference a new bottom boundary offset in which $(u, v) \in F$.

We now present the *find_pseudo_source_node algorithm* to tolerate at most three faulty nodes.

find_pseudo_source_node algorithm(){

if ($\exists(LB(u,v), BB(u,v)) \in R$ OR $\exists LB(u,v) == 0$ OR $\exists BB(u,v) == 0$, where $(u,v) \in F$)
{
set $x_{offset} = 0$ and $F1 = \{\}$;
while ($x_{offset} \leq 4$) {
if ($\exists LB(u,v) == x_{offset}$, where $(u,v) \in F$) {
$x_{offset} = x_{offset} + 1$;
continue; /* start another while loop for x_{offset} */
}
if ($x_{offset} == 0$)
set $y_{offset} = 1$;
else
set $y_{offset} = 0$;
while ($y_{offset} \leq 4$) {
if ($\exists BB(u,v) == y_{offset}$, where $(u,v) \in F$) {
$y_{offset} = y_{offset} + 1$;
continue; /* start another while loop for y_{offset} */
}
let $W = \{(i,j) \mid (RDO(LB(u,v), x_{offset}), RDO(BB(u,v), y_{offset})), where (u,v) \in F\}$;
if ($\forall (i,j) \in W$, $(i,j) \notin R$)
goto FOUND;
else if (the number of faults == 3)
$F1 = F1 + \{(x + x_{offset}, y + y_{offset})\}$;
$y_{offset} = y_{offset} + 1$;
} /* the end of while loop for y_{offset} */
$x_{offset} = x_{offset} + 1$;
} /* the end of while loop for x_{offset} */
select one element from set $F1$ as the pseudo-source node (x', y') and stop;
FOUND:
set the pseudo-source node $(x', y') = (x + x_{offset}, y + y_{offset})$;
} /* the end of if-then-else statement */
}

The *find_pseudo_source_node algorithm* selects a pseudo-source node such that the faulty node will neither be the branch node positions of any square grids partitioned by the new embedded

torus G' nor locate onto any edges or nodes of G'. Notice that by applying the simple greedy algorithm mentioned before, only one broadcasting step is required to transmit the message from the originator to the selected pseudo-source node.

Lemma 1 *A non-faulty pseudo-source node can always be found in the* 5×5 *mapping square grid with the originator located at the left-bottom corner of the grid by applying the* find_pseudo_source_node algorithm *in the presence of* r $(1 \leq r \leq 3)$ *faulty nodes such that*

1. *No faulty node is located on the new embedded torus* G'.
2. *When* r $(1 \leq r \leq 2)$ *faulty nodes exist, no branch nodes of basic group units with respect to* G' *are faulty.*
3. *When* $r = 3$ *faulty nodes exist, in the worst case only one branch node in a basic group unit with respect to* G' *can be faulty.*

Proof: Assume the position of the selected pseudo-source node is (i, j) and the new embedded torus is denoted G', respectively.

Part (1): Every time the *find_pseudo_source_node algorithm* selects a possible candidate of the pseudo-source node from the rest 24 positions in the mapping square grid, the algorithm checks that whether the faulty nodes are on the links/nodes of the new embedded torus G'. In the worst case when three faulty nodes are all located on different rows and columns with respect to the mapping square grid, there are still 4 out of 25 positions can be selected as the pseudo-source node (i, j) after eliminating all the possible nodes in the same column or row of the faulty nodes. So we can always select a pseudo-source node (i, j) to form a fault-free embedded torus G'.

Part (2): Consider the case that at most two faulty nodes are detected. From part (1) we know that in the worst case here only 9 out of 25 possible positions in the mapping square grid can be selected as the pseudo-source node (i, j). From the observation of Fig. **??**, we learn that the 9 possible positions are always distributed into 5 different groups. If a faulty node is mapping to one position in the same group (5 nodes) of the node (i, j), this indicates one branch node of a basic group unit with respect to the G' is faulty. In the worst case that two faulty nodes belong to two different groups we still have the available nodes of $5 - 2 = 3$ groups from which the pseudo-source node (i, j) can be selected. Therefore, a suitable (i, j) can always be selected to form a G' and no branch nodes of basic group units with respect to the G' are faulty.

Part (3): Now consider the case where 3 faulty nodes exist. From part (1) we know that only 4 possible positions in the mapping square grid are available to be selected. From the observation of Fig. **??** we know that the 4 possible positions are always distributed into 3 different groups.

That means in the worst case we cannot find a group without a faulty node located on a branch node position. □

4.1 Fault Tolerant Algorithm for Two Faults

The 2-*Fault Tolerant PSB Algorithm* for 2-dimensional torus is shown in the following.

2-Fault Tolerant PSB Algorithm(){

1. Construct the embedded torus graph G from the original source node (x, y).
2. Apply the *find_pseudo_source_node algorithm.*
3. Apply the *PSB algorithm* and stop.

}

Theorem 2 *In the presence of* r $(1 \leq r \leq 2)$ *faulty nodes at most one additional step is required by the* 2-Fault Tolerant PSB Algorithm *to perform a one-to-all broadcast in a* $5^p \times 5^p$ *circuit-switched torus network.*

Proof: The possible additional steps of the 2-*Fault Tolerant PSB Algorithm* compared to the *PSB Algorithm* are a result of the *find_pseudo_source_node algorithm.* From lemma **??** it is observed that at most one broadcasting step is performed by the *find_pseudo_source_node algorithm.* □

4.2 Fault Tolerant Algorithm for Three Faults

We now describe in detail the 3-*Fault Tolerant PSB Algorithm* to tolerate at most three node failures.

3-Fault Tolerant PSB Algorithm(){

1. Construct the embedded torus graph G from the original source node (x, y).
2. Apply the *find_pseudo_source_node algorithm.*
3. If the pseudo-source node $(x', y') \in F1$, apply the *Prebranch_Broadcast PSB algorithm* and stop. Otherwise, goto STEP 4.
4. Apply the *PSB algorithm* and stop.

}

Here a new algorithm called the *Prebranch_Broadcast PSB algorithm* is introduced to resolve the broadcast problem that one branch node of a basic group unit in torus T is faulty after selecting a pseudo-source node. Fig. **??**, **??**, and **??** profile this algorithm in detail. One basic group unit is outlined by the dotted line with the center root node labeled H. All black nodes indicate the branch nodes in the basic group unit. The node F is the faulty branch node and node P is the

corresponding pre-branch node of node F. Three leaf nodes of the branch node F are labeled A, B, and C. All gray nodes represent the possible positions of the two other faulty nodes. Notice that this condition only occurs when no two faults are located on the same row or the same column of torus T. Leaf nodes A, B, and C must be fault-free because of the position of the faulty branch node F. This guarantees a fault-free path can be established from the three default routing paths shown in Fig. ??, ??, and ?? to send the message from the pre-branch node P to the leaf node C in one broadcasting step because two faulty nodes at different row and column positions cannot break all three default routing paths simultaneously. At the same time the pre-branch node P sends the message to the leaf nodes A and B through the L-type paths also.

When compared to the *PSB algorithm*, the *Prebranch_Broadcast PSB algorithm* requires the same number of broadcasting steps. The only difference is that every node in the torus T must load the information of those predefined relative routing paths before applying the *Prebranch_Broadcast PSB algorithm*. The *Prebranch_Broadcast PSB algorithm* is presented in the following.

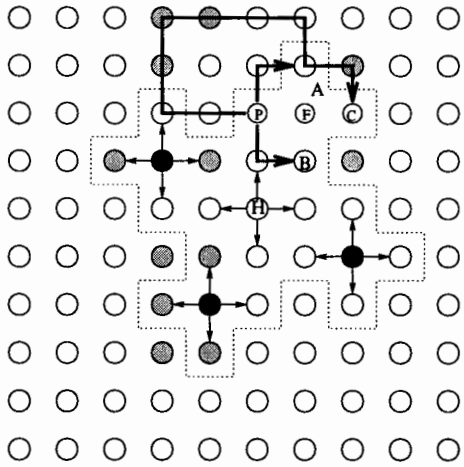
Prebranch_Broadcast PSB Algorithm(){

1. Apply the *PSB algorithm* at the first $2p-2$ broadcast steps in a $5^p \times 5^p$ torus.
2. Send the message from each root node of a basic group unit to the corresponding fault-free branch nodes. If a branch node is faulty, then send the message to its corresponding pre-branch node.
3. In the last broadcast step if the message is in a branch node, then apply the *PSB algorithm* and stop. Otherwise, goto STEP 4.
4. Apply $L-$type routing paths to transmit the message from the pre-branch node P to leaf nodes A and B. At the same time observe and select one fault-free routing path out of three possible default paths to broadcast the message to the leaf node C.

}

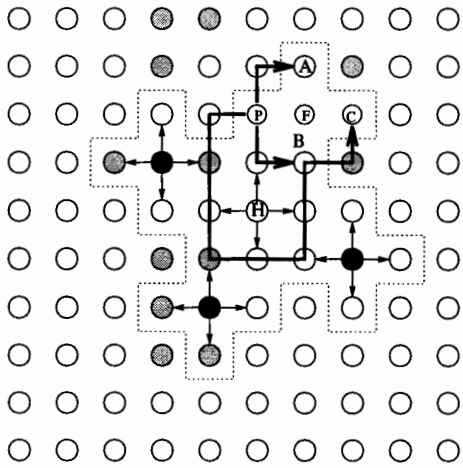
Theorem 3 *In the presence of* r $(1 \leq r \leq 3)$ *faulty nodes at most* 1 *additional step is required by the* 3-Fault Tolerant PSB Algorithm *to perform a one-to-all broadcast in a* $5^p \times 5^p$ *circuit-switched torus network.*

Proof: Consider the worst case: three faulty nodes exist in the torus network and the *Prebranch_Broadcast PSB algorithm* is executed. From lemma ?? it is proved that if necessary, we always can find and select a pseudo-source node by applying the *find_pseudo_source_node algorithm* and only one step is needed to transmit the message from originator to the selected pseudo-source node. Since the *Prebranch_Broadcast PSB algorithm* does not need any extra steps to broadcast a message when it is compared to the *PSB algorithm*, we can conclude that at most 1 additional step is sufficient for the 3-*Fault Tolerant PSB Algorithm* to perform a one-to-all broadcast in a $5^p \times 5^p$ circuit-switched torus network. □



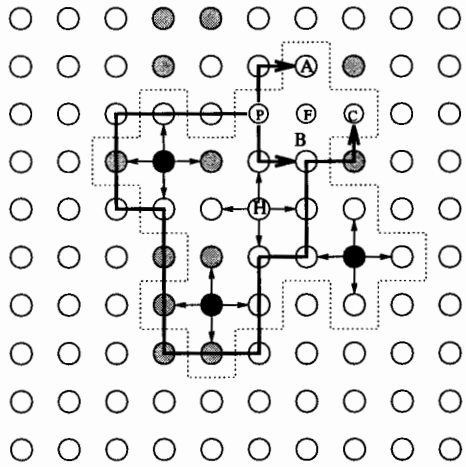
(a)

Figure 8: The first possible routing path from the pre-branch node P which substitutes its corresponding branch node F to broadcast the message to the leaf node C.



(b)

Figure 9: The second possible routing path.



(c)

Figure 10: The third possible routing path.

Theorem 4 *The* 3-Fault Tolerant PSB Algorithm *is an optimal fault tolerant algorithm compared to the* PSB algorithm.

Proof: Assume the originator is (x, y) and one of the four neighbor nodes of (x, y) is faulty. After applying the *PSB algorithm* one fourth of the total number of nodes in the 2-D torus network cannot receive the broadcast message from node (x, y). It is obvious that at least one additional broadcast step is required to allow one fourth of the nodes to receive the message in the presence of one faulty node. Because the 3-*Fault Tolerant PSB Algorithm* only needs one more broadcast step to tolerate up to 3 node failures while compared to the *PSB algorithm*, the 3-*Fault Tolerant PSB Algorithm* is optimal. □

5 Conclusion

In this paper we have derived a fault tolerant broadcasting algorithm for a $5^p \times 5^p$ 2-dimension circuit-switched torus network to tolerate up to 3 permanent node failures. Because there is at most one additional step needed in our algorithm to accomplish a broadcasting task compared to the optimal broadcasting algorithm derived from Peters and Syska [?], our fault tolerant algorithms are optimal and the complexity of this algorithm is $O(\log_5 N)$ in which N is the total number of nodes in a $2-$D torus network.

The future work is to consider the multiple node (> 3) failure problem and the dynamical link/node failure problem. Both problems can form an incomplete 2-dimensional torus with complex irregular shape. It will also be interesting to develop fault tolerant algorithms on n-dimensional torus network in the presence of at most $2n - 1$ faults in which $n \geq 3$.

References

[1] Joseph G. Peters, and Michel Syska. "Circuit-Switched Broadcasting in Torus Networks," *IEEE Tran. on Par. and Dis. Sys.*, vol. 7, NO. 3, pp. 246-255, 1996.

[2] Ju-Young L. Park, and Hyeong-Ah Choi. "Circuit-Switched Broadcasting in Torus and Mesh Networks," *IEEE Tran. on Par. and Dis. Sys.*, vol. 7, NO. 2, pp. 184-190, 1996.

[3] Yih-jia Tsai, and Philip K. McKinley. "A Broadcast Algorithm for All-Port Wormhole-Routed Torus Networks", *IEEE Tran. on Par. and Dis. Sys.*, vol. 7, NO. 8, pp. 876-885, 1996.

[4] Ju-Young L. Park, Sang-Kyu Lee, and Hyeong-Ah Choi. "Fault-Tolerant Broadcasting in Circuit-Switched Mesh", *Proc. SIAM Conf. on Parallel Processing for Scientific Computing*, pp. 22-24, Mar., 1993.

[5] W.J. Dally, and C.L. Seitz. "Deadlock-free Message Routing in Multiprocessor Interconnection Networks", *IEEE Tran. on Computers*, vol. 36, NO. 5, pp. 547-553, 1987.

[6] S. H. Bokhari. "A Network Flow Model for Load Balancing in Circuit-Switched Multicomputers", *ICASE Report* 90-38, May 1990.

[7] S. H. Bokhari. "Communication Overheads on the Intel iPSC-2 Hypercube", *ICASE Interim Report* 10, May 1990.

[8] P. Fraigniaud, and E. Lazard. "Methods and Problems of Communication in Usual Networks", *Discrete Applied Math.*, vol. 53, pp. 79-133, 1994.

[9] D.K. Pradhan. *Fault Tolerant Computing*, Prentice/Hall International, Englewood Cliffs, New Jersey, 1986.