Fault-Tolerant Broadcasting in 2-D Wormhole-Routed Meshes *

Zhen Jiang and Jie Wu
Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431

Abstract

In this paper, a fault-tolerant broadcast scheme in 2-D meshes with randomly generated faults is provided. This approach is based on an early work on time-step optimal broadcasting in square-shape fault-free 2-D meshes with optimal *total communication distance* (TCD). An extension to any rectangular-shape fault-free 2-D meshes is first given. The fault block model is used in which all faulty nodes in the system are contained in a set of disjoint blocks. The boundary lines of blocks divide the whole mesh into a set of fault-free polygons and a sequence of rectangular fault-free regions is derived from these polygons. The broadcast process is carried out at two levels: inter-region and intra-region. In the inter-region-level broadcast, the broadcast message is sent from a given source to a special node (called *eye*) in each rectangular fault-free region. In the intra-region-level broadcast, the extended optimal fault-free broadcast is applied. Some analytical results are given including an upper bound of TCD.

Keywords: *Broadcast, communication distance, fault tolerance, meshes, wormhole routing*

*This work was supported in part by NSF grant CCR 9900646. Email:{zjiang, jie}@cse.fau.edu.

1 Introduction

In a multicomputer system, a collection of processors (also called nodes) work together to solve large application problems. These nodes communicate and coordinate their efforts by sending and receiving messages through the underlying communication network. Thus, the performance of such a multicomputer system is dependent on the end-to-end cost of communication mechanisms.

Minimizing communication latency is important for an efficient implementation of collective communication operations [4, 5] which include multicast and broadcast. *Broadcast* [2] is a special case of multicast in which the same message is delivered to all the nodes. Broadcast is essential in many applications such as distributed agreement [3], clock synchronization [6], and compute-aggregate-broadcast type of algorithms [1].

We assume that the system under consideration uses the one-port model; that is, at each time step a node may perform one of the following operations: send a message to one node, receive a message from one node, or stay in idle. Under the wormhole switching, forwarding a message from one node to any other node is considered as one time step which is irrelevant to the distance between these two nodes, provided there is no traffic contention.

In the wormhole-routed system, *traffic contention* includes step contention and depth contention. *Step contention* occurs when two copies of a message in the same time step contend for a common channel. Another contention is called *depth contention* that is defined as two copies of a message in different time steps contend for a common channel. This situation occurs if the broadcast message is long or one of the copies is delayed and transmitted at a later step. This paper focuses on avoiding step contention. Depth contention is not considered, assuming that the broadcast message is relatively short.

The traffic in such a system can be measured by *total communication distance* (TCD) which is the summation of all the distances a broadcast message traverses during the broadcast process. Obviously, the overall network traffic contention, as well as the communication delay, depends on the TCD. Therefore, minimizing the TCD is important in designing an efficient broadcast. Note that without the minimum TCD requirement, time-step broadcasting can be easily achieved through *recursive doubling*; that is, the number of nodes that receive a copy of the message doubles after each step. The challenge here is to generate a routing path that guarantees a minimum TCD without traffic contention at any time step.

Wu and Cang [1] showed that from a special node (called *eye*) in a $2^k \times 2^k$ mesh, the time-step optimal broadcast that always forwards the broadcast message to several fixed locations in

a predefined order achieves an optimal TCD. Note that optimal TCD is globally minimum TCD regardless of the location of the source.

When the shape of the mesh changes, the locations of eyes also change. In this paper, an extension to any rectangular-shape fault-free 2-D meshes is first given together with a new definition for eyes based on Wu and Cang's optimal TCD broadcasting [1]. For an $m \times n$ mesh with randomly generated faults, a fault block model is used in which all faulty nodes are contained in a set of disjoint blocks. The boundary lines of blocks divide the whole mesh into a set of fault-free polygons and a sequence of rectangular fault-free regions is derived from these polygons in a column-major form (from the west-most column to the east-most column). Subsequent broadcasting within each rectangular fault-free region does not interfere with communication in other rectangular regions. Therefore, traffic contentions are avoided. The fault-tolerant broadcast process is carried out at two levels: inter-region and intra-region. In the inter-region-level broadcast, the broadcast message is sent from a given source to a special node (eye) in each region. In the intra-region-level broadcast, the extended optimal fault-free broadcast is applied from the eye within each region.

Given a 2-D mesh with fault blocks, it is difficult to design an optimal TCD broadcast from an eye. A general method optimized for any fault distribution is also impractical. Our study thus focuses on a simple broadcast algorithm and its performance is bounded in terms of TCD costs. The analytical results show that our algorithm can complete a broadcast in $1 + \lceil \lg m \rceil + \lceil \lg n \rceil + \lceil \lg(3f + 1) \rceil$ steps in a faulty $m \times n$ mesh, where f is the number of fault blocks, compared with $\lceil \lg m \rceil + \lceil \lg n \rceil$ steps in a fault-free $m \times n$ mesh. In the subsequent discussion, each rectangular-shape fault-free region is simply called a region.

The remainder of the paper is organized as follows. Section 2 introduces necessary notations and preliminaries. The concept of eye and the optimal TCD broadcast in a $2^k \times 2^k$ fault-free 2-D mesh are reviewed. Section 3 defines the eye in an $m \times n$ rectangular fault-free 2-D mesh and extends the optimal TCD broadcast algorithm to such a mesh. Section 4 presents the two-phase fault-tolerant broadcast without traffic contention. This broadcast includes region division, inter-region-level broadcast, and intra-region-level broadcast. The analytical results and examples show the scalability of such a broadcast. Section 5 concludes the paper and provides ideas for future research.

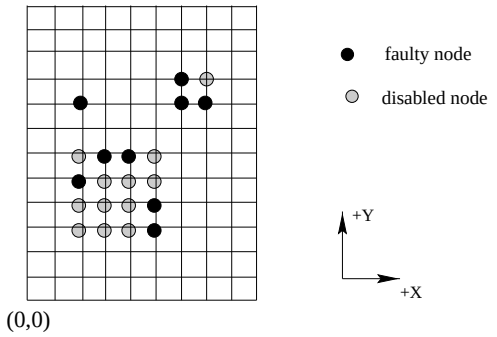

Figure 1: Distinguishing disabled or faulty nodes along different dimensions.

2 Preliminaries

2.1 Meshes and block fault model

A k-ary n-dimensional (n-D) mesh with $k_1 \times k_2 \times ... \times k_n$ nodes has an interior node degree of $2n$ and the network diameter is $(k-1)n$. Each node u has an address $(u_1, u_2, ..., u_n)$, where $u_i = 0, 1, ..., k_i - 1$. Two nodes $(v_1, v_2, ..., v_n)$ and $(u_1, u_2, ..., u_n)$ are connected if their addresses differ in one and only one dimension, say dimension i; moreover, $|v_i - u_i| = 1$. Basically, nodes along each dimension are connected as a linear array. Each node u in a 2-D mesh is labeled as (x_u, y_u) or simply (x, y).

Most existing literatures on fault-tolerant routing use disjoint rectangular blocks to model node faults and to avoid routing difficulties in meshes. First, a node-labeling scheme is given that either enabled or disabled is assigned to each non-faulty node. Adjacent disabled and faulty nodes form a faulty rectangle. Such a rectangle is called a rectangular fault block, or simply fault block.

Definition 1: *In a 2-D mesh a non-faulty node is initially labeled enabled; however, its status will be changed to disabled if there are two or more disabled or faulty neighbors in different dimensions. Connected disabled and faulty nodes form a fault block.*

Figure 1 shows a 2-D mesh with nine faults (2,5), (2,8), (3,6), (4,6), (5,4), (5,3), (6,8), (6,9), and (7,8). The corresponding fault blocks are [2:5, 3:6], [2:2, 8:8], and [6:7,8:9]. The block fault model has the following interesting property: In a 2-D mesh, each fault block is a rectangle and the distance between any two fault blocks is at least two [8]. To simplify the discussion, we assume that there is no fault on the edges of the mesh.

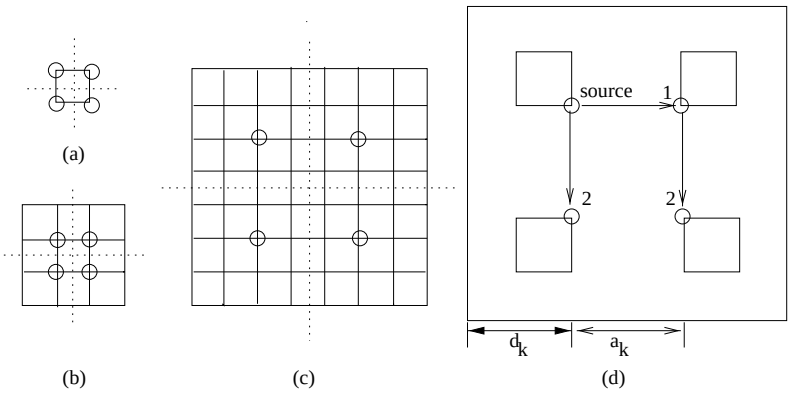

Figure 2: The recursive definition of eyes of (a) a 2×2 mesh, (b) a 4×4 mesh, (c) an 8×8 mesh, and (d) a $2^k \times 2^k$ mesh.

2.2 Optimal TCD Broadcast in $2^k \times 2^k$ meshes

There are four eyes in a $2^k \times 2^k$ mesh with $k \geq 1$, labeled as E_k^2. These eyes are recursively defined as follows: All four nodes in a 2×2 mesh are eyes (see Figure 2(a)). A $2^k \times 2^k$ mesh is partitioned into four $2^{k-1} \times 2^{k-1}$ submeshes, each of which has four eyes. E_k^2 are selected from sixteen E_{k-1}^2s. Specifically, E_k^2s of the upper-left, upper-right, lower-left, and lower-right submeshes are four E_{k-1}^2s that are the closest to the center of the $2^k \times 2^k$ mesh among the sixteen E_{k-1}^2s, as shown in Figure 2(d).

For example, the inner four nodes of a 4×4 mesh as shown in Figure 2(b) are eyes. Figure 2(c) shows four eyes of an 8×8 mesh. Denote a_k as the length of the side of eye-square in a $2^k \times 2^k$ mesh and d_k as the distance of the eye from the edge of this mesh. a_k and d_k are calculated in [1] by

$$a_k = 2^{k-1} - a_{k-1}, \tag{1}$$

$$d_k = \frac{1}{2}[2^k - 1 - a_k], \quad k \geq 2. \tag{2}$$

where $a_1 = 1$ and $d_1 = 0$. This recursive formula leads to

$$a_k = \frac{1}{3}[2^k - (-1)^k], \tag{3}$$

$$d_k = \frac{1}{6}[2^{k+1} + (-1)^k] - \frac{1}{2}, \quad k \geq 1. \tag{4}$$

We can easily determine locations of the first four eyes of a given $2^k \times 2^k$ mesh as: $(d_k,\ d_k)$, $(2^k - 1 - d_k,\ d_k)$, $(d_k,\ 2^k - 1 - d_k)$, and $(2^k - 1 - d_k,\ 2^k - 1 - d_k)$ (see Figure 2(d)).

Algorithm 1: Optimal TCD broadcast algorithm in a $2^k \times 2^k$ mesh with $k \geq 1$

1. Divide the given $2^k \times 2^k$ mesh into four $2^{k-1} \times 2^{k-1}$ submeshes. Rotate the mesh, if necessary, until source node (one of E_k^2s) is in the upper-left submesh.
2. The source node sends the message to the upper-right eye of E_k^2s in the first step.
3. In the second step, the source and the upper-right eye send the message to the lower eyes.
4. In the remaining steps, the four submeshes deliver the message within their own submeshes of the next level following the above procedure. In this way the message is delivered down to the submeshes level by level until reaching unit 2×2 meshes, and all these unit meshes complete the broadcast process within themselves in two steps.

For example, $d_1 = 0$, $d_3 = 2$, and $d_4 = 5$; similarly, $a_1 = 1$, $a_3 = 3$, and $a_4 = 5$. The coordinates of the first four eyes of a unit 2×2 mesh are (0,0), (1,0), (0,1), and (1,1). The coordinates of the first four eyes of an 8×8 mesh are (2,2), (5,2), (2,5), and (5,5). The coordinates of the first four eyes of an 16×16 mesh are (5,5), (10,5), (5,10), and (10,10).

Algorithm 1 shown in [1] is an optimal TCD broadcast algorithm in which the source is an eye of a $2^k \times 2^k$ mesh. Its optimal TCD in a $2^k \times 2^k$ mesh, OD_k^2, can be calculated by

$$OD_k^2 = \frac{1}{5}[3 \times 2^{2k+1} - (-1)^k] - 2^k. \tag{5}$$

For example, $OD_1^2 = 3$, $OD_3^2 = 69$, and $OD_4^2 = 291$. Wu and Cang [1] also gave a general minimum TCD broadcast in which the source is not an eye.

3 Extended Optimal TCD Broadcast

As we discussed early, a $2^k \times 2^k$ mesh is necessary condition for Algorithm 1. Here we extend Algorithm 1 to an $m \times n$ rectangular fault-free mesh with $m \geq n \geq 1$. The extended algorithm also starts from an eye and the message is delivered to all the nodes in such a rectangular mesh without contention. Although optimality no longer holds, in the extended algorithm, an upper bound of the corresponding TCD is given.

Algorithm 2: Extended optimal TCD broadcast algorithm in an $m \times n$ mesh with $m \geq n \geq 1$

1. Assume the source is E, one of the four eyes of a mesh $[0 : m-1, 0 : n-1]$.
2. The mesh is partitioned into $A : [0 : \lceil \frac{m}{2} \rceil - 1, 0 : n-1]$ and $B : [\lceil \frac{m}{2} \rceil : m-1, 0 : n-1]$.
3. Source E sends the message to the closest eye E' out of four in submesh B. The source E is still an eye in submesh A.
4. In the remaining steps, the two submeshes A and B deliver the message within their own submeshes of the next level following the above procedure until reaching unit 1×1 meshes.

3.1 Eyes of a region

In an $m \times n$ fault-free mesh: $[0 : m-1, 0 : n-1]$, four eyes are defined as $E_0 : (D_m, D_n)$, $E_1 : (m-1-D_m, D_n)$, $E_2 : (D_m, n-1-D_n)$, and $E_3 : (m-1-D_m, n-1-D_n)$. D_m and D_n are defined by

$$D_k = \begin{cases} 0 & k = 1 \\ \lceil \frac{k}{2} \rceil - 1 - D_{\lceil \frac{k}{2} \rceil} & k > 1 \end{cases}$$

where k is either m or n.

Theorem 1: $\lfloor \frac{k-1}{3} \rfloor \leq D_k \leq \lfloor \frac{k+1}{3} \rfloor$ *for any* $k \geq 1$.

Proof: We prove the above claim by induction. When $k = 1$, $\lfloor \frac{1-1}{3} \rfloor \leq D_k = 0 \leq \lfloor \frac{1+1}{3} \rfloor = 0$. Assume the result holds for all $k \leq k'$. Consider $k = k' + 1$. $\lfloor \frac{k-1}{3} \rfloor \leq \lfloor \frac{k+1}{2} \rfloor - 1 - \lfloor \frac{\lceil \frac{k}{2} \rceil + 1}{3} \rfloor \leq D_k = \lceil \frac{k}{2} \rceil - 1 - D_{\lceil \frac{k}{2} \rceil} \leq \lfloor \frac{k+1}{2} \rfloor - 1 - \lfloor \frac{\lceil \frac{k}{2} \rceil - 1}{3} \rfloor \leq \lfloor \frac{k+1}{3} \rfloor$. Thus, $\lfloor \frac{k-1}{3} \rfloor \leq D_k \leq \lfloor \frac{k+1}{3} \rfloor$ for any $k \geq 1$. □

For example, $\lfloor \frac{2-1}{3} \rfloor \leq D_2 = 1 - 1 - D_0 = 0 \leq \lfloor \frac{2+1}{3} \rfloor$, $\lfloor \frac{5-1}{3} \rfloor \leq D_5 = 3 - 1 - D_3 = 3 - 1 - (2 - 1 - D_2) = 1 \leq \lfloor \frac{5+1}{3} \rfloor$, and $\lfloor \frac{7-1}{3} \rfloor \leq D_7 = 4 - 1 - D_4 = 4 - 1 - (2 - 1 - D_2) = 2 \leq \lfloor \frac{7+1}{3} \rfloor$. The coordinates of four eyes of a 7×5 mesh are $E_0 : (2, 1)$, $E_1 : (4, 1)$, $E_2 : (2, 3)$, and $E_3 : (4, 3)$.

The position of an eye in the $m \times n$ mesh is defined based on its distance to the edges of the mesh in two dimensions (D_m in the X dimension and D_n in the Y dimension). The base case is a 1×1 unit mesh: $[0 : 0, 0 : 0]$ $(m = n = 1)$ where all four eyes point to the same node. In $[0 : m-1, 0 : 0]$, there are two eyes $(D_m, 0)$ and $(m-1-D_m, 0)$.

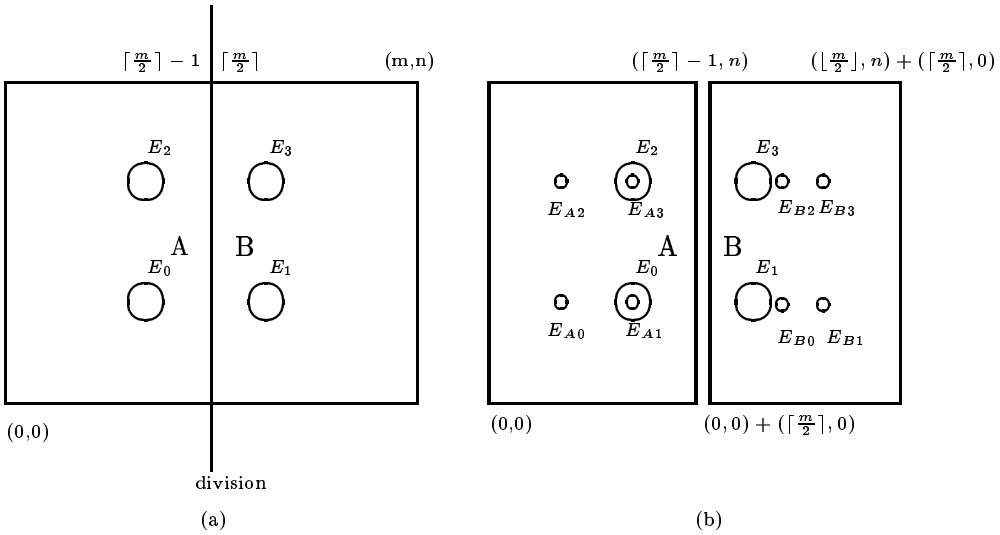

Figure 3: Eyes of different levels in a fault-free 2-D mesh.

3.2 Extended optimal TCD broadcast

The optimal TCD broadcast in Algorithm 1 is extended to an rectangular $m \times n$ fault-free mesh as shown in Algorithm 2. Assume that the source E is one of the four eyes (E_0, E_1, E_2, and E_3) of a mesh $[0 : m-1, 0 : n-1]$ with $m \geq n \geq 1$ as shown in Figure 3(a). The mesh is partitioned into $A : [0 : \lceil \frac{m}{2} \rceil - 1, 0 : n-1]$ and $B : [\lceil \frac{m}{2} \rceil : m-1, 0 : n-1]$. Source E sends a copy of the message to the closest eye E' out of four in submesh B. The source E is still an eye in submesh A. It is noted that E' is one of the four eyes in submesh B (one of E_{B0}, E_{B1}, E_{B2}, and E_{B3} as shown in Figure 3(b)) but may not be an eye before the division (one of E_0, E_1, E_2, and E_3). In the remaining steps, the two submeshes A and B deliver the message within their own submeshes of the next level following the above procedure. In this way, the message is delivered down to the submeshes level by level until reaching unit 1×1 meshes.

More specifically, in $[0 : m-1, 0 : n-1]$ there are four symmetrical eyes $E_0 : (D_m, D_n)$, $E_1 : (m-1-D_m, D_n)$, $E_2 : (D_m, n-1-D_n)$, and $E_3 : (m-1-D_m, n-1-D_n)$. Any of them can be selected as an eye to broadcast the message to all the nodes inside this region. Assume that $E_0 : (D_m, D_n)$ is the selected eye E. The whole mesh is partitioned into two submesh $A : [0 : \lceil \frac{m}{2} \rceil - 1, 0 : n-1]$ and $B : [\lceil \frac{m}{2} \rceil : m-1, 0 : n-1]$ by a half division. The distance D_m

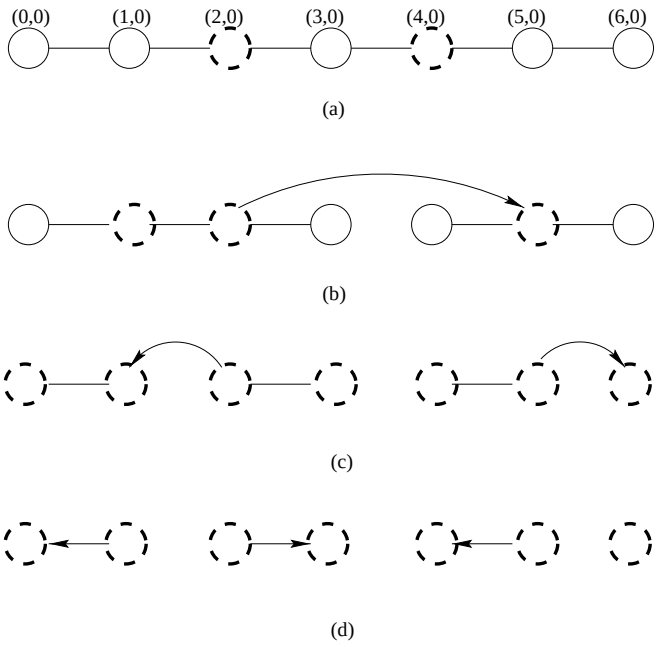

Figure 4: Broadcast in a 7×1 mesh.

ensures that the position of E is distance $D_{\lceil \frac{m}{2} \rceil}$ from the line $x = \lceil \frac{m}{2} \rceil - 1$; that is, it is also an eye of its submesh A (E_{A1} in Figure 3(b)). There are also four eyes in submesh B: E_{B0}, E_{B1}, E_{B2}, and E_{B3}. E_{B0} is the closest eye E' to E out of these four. Based on the definition of eye, $E_{B0} : (\lceil \frac{m}{2} \rceil + D_{\lfloor \frac{m}{2} \rfloor}, D_n)$ may not be $E_1 : (m - 1 - D_m, D_n)$ unless $\lfloor \frac{m}{2} \rfloor + D_{\lceil \frac{m}{2} \rceil} = \lceil \frac{m}{2} \rceil + D_{\lfloor \frac{m}{2} \rfloor}$. After source E sends a copy of the message to E' (see Figure 3(b)), the two submeshes A and B deliver the message within their own submeshes starting from E and E'. In this way, the message is delivered down to the submeshes level by level until reaching unit 1×1 meshes. The distance between current source E and the selected eye E', A_m, can be calculated by

$$A_m = \lceil \frac{m}{2} \rceil - D_m + D_{m - \lceil \frac{m}{2} \rceil} = 1 + D_{\lfloor \frac{m}{2} \rfloor} + D_{\lceil \frac{m}{2} \rceil}. \tag{6}$$

Theorem 2: $\lfloor \frac{m-1}{3} \rfloor \leq A_m \leq 1 + \lceil \frac{m}{3} \rceil$ *for any* $m \geq 1$.

Proof: Since $\lfloor \frac{\lceil \frac{m}{2} \rceil - 1}{3} \rfloor \leq D_{\lceil \frac{m}{2} \rceil} \leq \lfloor \frac{\lceil \frac{m}{2} \rceil + 1}{3} \rfloor$ and $\lfloor \frac{\lfloor \frac{m}{2} \rfloor - 1}{3} \rfloor \leq D_{\lfloor \frac{m}{2} \rfloor} \leq \lfloor \frac{\lfloor \frac{m}{2} \rfloor + 1}{3} \rfloor$ (based on Theorem 1), $\lfloor \frac{m-1}{3} \rfloor \leq 1 + \lfloor \frac{\lfloor \frac{m}{2} \rfloor - 1}{3} \rfloor + \lfloor \frac{\lceil \frac{m}{2} \rceil - 1}{3} \rfloor \leq A_m = 1 + D_{\lfloor \frac{m}{2} \rfloor} + D_{\lceil \frac{m}{2} \rceil} \leq 1 + \lfloor \frac{\lfloor \frac{m}{2} \rfloor + 1}{3} \rfloor + \lfloor \frac{\lceil \frac{m}{2} \rceil + 1}{3} \rfloor \leq 1 + \lceil \frac{m}{3} \rceil$. □

For example, $A_{22} = 9 \leq 9$ and $A_{189} = 62 \leq 64$. Figure 4 shows an example of our extended broadcast in a 7×1 mesh, where the dotted cycles are eyes at the level under consideration. $(2, 0)$ and $(4, 0)$ are the first two eyes in $[0 : 6, 0 : 0]$ (see Figure 4(a)). Assume that $(2, 0)$ is the source.

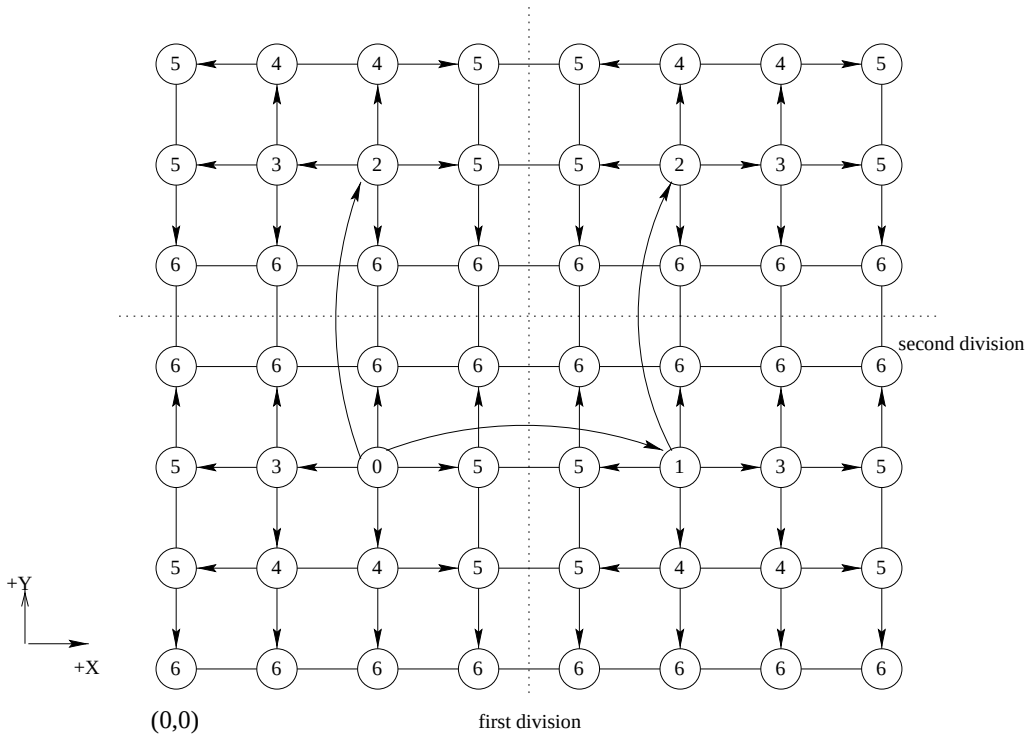

Figure 5: Broadcast in a 8×7 mesh.

The mesh is partitioned into $A : [0:3, 0:0]$ and $B : [4:6, 0:0]$. $(1,0)$ and $(2,0)$ are the eyes of the next level in submesh A and $(5,0)$ is the only eye of the next level in submesh B (see Figure 4(b)). After $(2,0)$ sends a copy of the message to $(5,0)$, the division in submesh A generates $[0:1, 0:0]$ and $[2:3, 0:0]$ (see Figure 4(c)). And $(2,0)$ will send a copy of the message to the closest eye of the next level in $[0:1, 0:0]$: $(1,0)$. At the same time step, submesh B is partitioned into $[4:5, 0:0]$ and $[6:6, 0:0]$. $(5,0)$ will send a copy of the message to the eye of the next level $(6,0)$. In the next time step (see Figure 4(d)), eyes $(1,0)$, $(2,0)$, and $(5,0)$ of submeshes $[0:1, 0:0]$, $[2:3, 0:0]$ and $[4:5, 0:0]$ send copies of the message to the closest eyes of the next level $(0,0)$, $(3,0)$, and $(4,0)$ of $[0:0, 0:0]$, $[3:3, 0:0]$, and $[4:4, 0:0]$, respectively.

Figure 5 shows an example of our extended broadcast in a 8×7 mesh, where $(2,2)$ is one of the eyes in this mesh. Assume that $(2,2)$ is the source. The mesh is partitioned into $[0:3, 0:6]$ and $[4:7, 0:6]$ since $m > n$. There are four eyes in the 4×7 submesh $[4:7, 0:6]$: $(5,2)$, $(6,2)$, $(5,5)$, and $(6,5)$. The source sends a copy of the message to the closest one: $(5,2)$. In the second step, the mesh is partitioned into $[0:3, 0:3]$, $[0:3, 4:6]$, $[4:7, 0:3]$, and $[4:7, 4:6]$. $(2,2)$ sends a copy to $(2,5)$ and $(5,2)$ sends a copy to $(5,5)$. Based on Algorithm 2, eye $(2,2)$ in $[0:3, 0:3]$ selects $(1,2)$

as the forwarding node in the third step. Submesh $[0:3,4:6]$ is partitioned into $[0:1,4:6]$ and $[2:3,4:6]$, and then, $(2,5)$ sends a copy to $(1,5)$ in the third step. In the fourth step, a submesh $[0:1,4:6]$ is partitioned into $[0:1,4:5]$ and $[0:1,6:6]$ and $(1,5)$ sends a copy of the message to $(1,6)$. The number in each node (a circle) represents the step that a copy of the message arrives.

Algorithm 2 is not an optimal TCD broadcast algorithm in an $m \times n$ rectangular fault-free mesh. However, for each step, its TCD is no more than that of a $2^k \times 2^k$ mesh as shown in Theorem 3, where k is $\max\{\lceil \lg m \rceil, \lceil \lg n \rceil\}$.

Lemma 1: *$D_m \geq D_{m'}$ and $A_m \geq A_{m'}$ if $m \geq m' \geq 1$.*

Proof: First, we prove by induction that for any $k \geq 1$, $D_k+1 \geq D_{k+1}$ and $D_k \leq D_{k+1}$. When $k = 1$, $D_1 = D_2 = 0$. $D_1+1 > D_2$ and $D_1 \leq D_2$. Assume the result holds for all $k \leq k'$. Consider $k = k'+1$. If k is even ($k = 2l$), $D_k+1 = l-1-D_l+1 \geq l-D_{l+1} = D_{k+1}$ and $D_k = l-1-D_l \leq l-D_{l+1} = D_{k+1}$; otherwise, k is odd ($k = 2l+1$), $D_k+1 = l+1-D_{l+1} > l-D_{l+1} = D_{k+1}$ and $D_k = l-D_{l+1} = D_{k+1}$. Thus, $D_m \geq D_{m'}$ if $m \geq m' \geq 1$.

Since $A_m = 1 + D_{\lfloor \frac{m}{2} \rfloor} + D_{\lceil \frac{m}{2} \rceil}$, $A_{m'} = 1 + D_{\lfloor \frac{m'}{2} \rfloor} + D_{\lceil \frac{m'}{2} \rceil}$, and based on the above result $D_{\lfloor \frac{m}{2} \rfloor} \geq D_{\lfloor \frac{m'}{2} \rfloor}$ and $D_{\lceil \frac{m}{2} \rceil} \geq D_{\lceil \frac{m'}{2} \rceil}$ if $m \geq m' \geq 1$. Therefore, $A_m \geq A_{m'}$. □

Theorem 3: *The TCD of the extended optimal broadcasting in an $m \times n$ mesh, denoted as $ED_{m \times n}$, can be limited by*

$$ED_{m \times n} \leq OD_k^2 = \frac{1}{5}[3 \times 2^{2k+1} - (-1)^k] - 2^k$$

where k is $\max\{\lceil \lg m \rceil, \lceil \lg n \rceil\}$.

Proof: Let $ED_{m \times n}$ be the TCD in an $m \times n$ mesh based on the proposed algorithm. We should prove that $ED_{m \times n} \geq ED_{m' \times n'}$ for all $m \geq m' \geq 1$, $n \geq n' \geq 1$. We prove the above claim by induction on $m + n$.

When $m + n = 2$, there is only choice: $m = n = m' = n'$. $ED_{1 \times 1} \geq ED_{1 \times 1}$. Assume the result holds for all $m + n \leq k'$. Consider $ED_{m \times n}$ and $ED_{m' \times n'}$ where $m + n = k' + 1$. Based on the partition process in Algorithm 2, the $m \times n$ mesh ($m' \times n'$ mesh) is partitioned into two submeshes: $\lceil \frac{m}{2} \rceil \times n$ and $\lfloor \frac{m}{2} \rfloor \times n$ ($\lceil \frac{m'}{2} \rceil * n'$ and $\lfloor \frac{m'}{2} \rfloor * n'$). Therefore, $ED_{m \times n} = A_m + ED_{\lceil \frac{m}{2} \rceil \times n} + ED_{\lfloor \frac{m}{2} \rfloor \times n}$ and $ED_{m' \times n'} = A_{m'} + ED_{\lceil \frac{m'}{2} \rceil \times n'} + ED_{\lfloor \frac{m'}{2} \rfloor \times n'}$. By our induction assumption and Lemma 1, $ED_{m \times n} \geq ED_{m' \times n'}$. If $m = n = 2^k$, $D_m = D_n = d_k = D_{2^k}$ and Algorithm 2 is the same as Algorithm 1; and hence, $ED_{m \times n} = OD_k^2$. If $m, n \leq 2^k$, $ED_{m \times n} \leq ED_{2^k \times 2^k} = OD_k^2$. □

For example, $ED_{7 \times 8} = 61 \leq OD^2_{\lceil \lg 8 \rceil} = 69$. Note that Theorem 3 just gives a simple approximation for $ED_{m \times n}$. In [9], a tighter approximation is given in which $ED_{m \times n}$ is represented as a

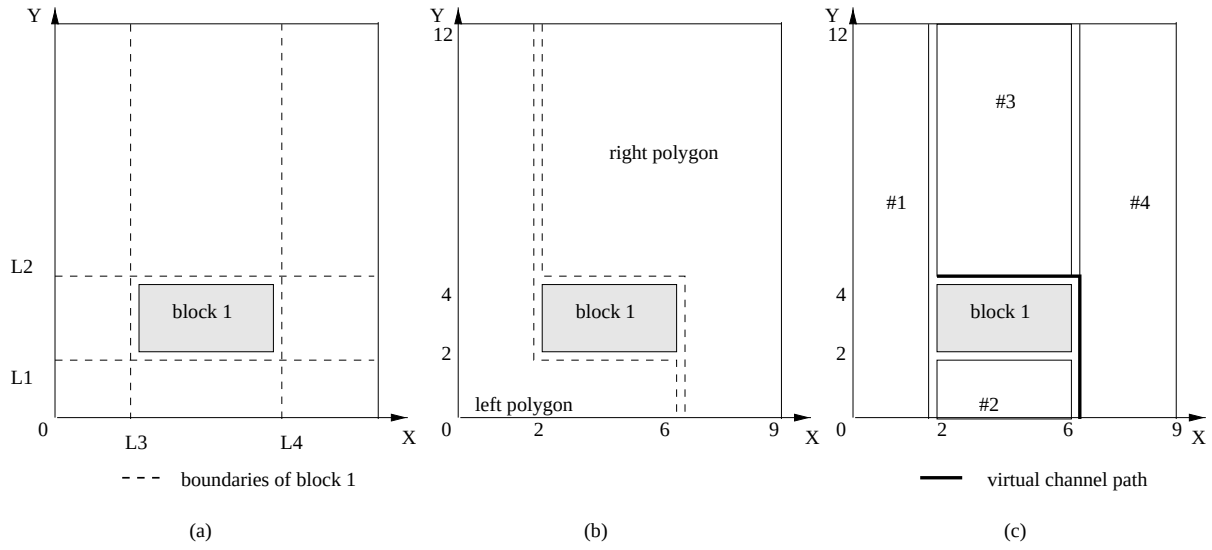

Figure 6: Region division in a mesh with one fault block.

function of k_1 and k_2, where $2^{k_1} \leq m < 2^{k_1+1}$ and $2^{k_2} \leq n < 2^{k_2+1}$.

4 Fault-Tolerant Broadcast

Consider a mesh with a set of disjoint fault blocks as defined by Definition 1. For a contention-free broadcast, a mesh is partitioned into a set of fault-free polygons, and then, a sequence of rectangular fault-free regions is derived from these polygons in a column-major form (from the west-most column to the east-most column). Once regions are defined, inter-region-level broadcast is applied, followed by intra-region-level broadcast.

In [7], Wu defined four boundary lines for each fault block. Let L_1, L_2, L_3, and L_4 correspond to south, north, west, and east boundary lines, respectively. Figure 6(a) shows an example of boundaries ($L_1 : y = 1$, $L_2 : y = 5$, $L_3 : x = 1$, and $L_4 : x = 7$) of block $[2 : 6, 2 : 4]$.

In Figure 6(b), together with the upper section of L_3 and the lower section of L_4, a fault block divides a mesh into two fault-free *orthogonal convex polygons* [8], or simply, fault-free polygons. One is called left polygon and the other right polygon. An area is *orthogonal convex* if and only if the following condition holds: For any horizontal or vertical line, if two nodes on the line are inside the area, all the nodes on the line that are between these two nodes are also inside the area. Figure 6(b) shows a partition $[0 : 1, 0 : 12] \cup [2 : 6, 0 : 1]$ and $[2 : 6, 5 : 12] \cup [7 : 9, 0 : 12]$ by

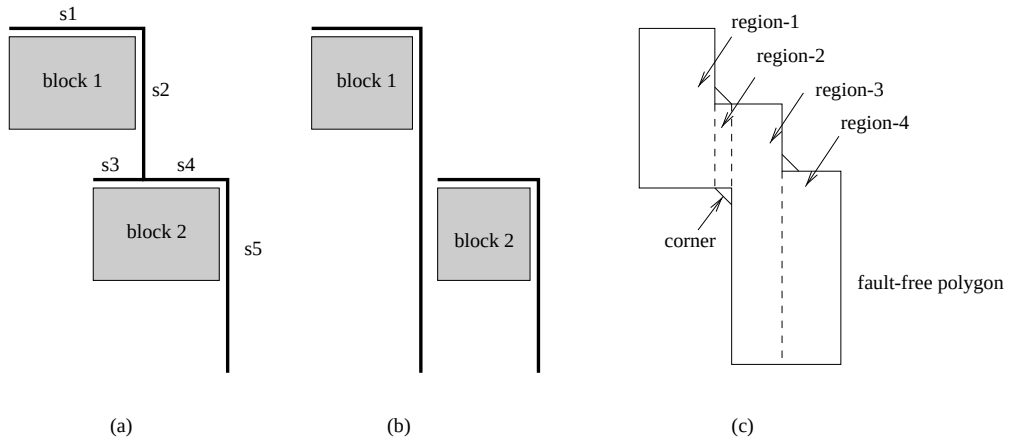

Figure 7: (a) Virtual channel path of two intersected fault blocks, (b) virtual channel path of two independent fault blocks, and (c) partition a polygon into a sequence of regions.

fault block $[2:6, 2:4]$. A set of *extra virtual channels* are used in the lower section of L_4 and the adjacent edges of fault block in the right polygon of the partition (see Figure 6(c)). For each physical channel where the notion of virtual channel is used, there are two virtual channels. One of them is used to form a *virtual channel path* (see the thick line in Figure 6(c)) to avoid contention of two packets competing the same channel (as will be discussed later). The virtual channel paths of two intersected fault blocks can be combined. For example, in Figure 7(a), there are two virtual channel paths: s_1-s_2-s_4-s_5 and s_3-s_4-s_5.

For each fault block, a mesh is partitioned into two polygons. These polygons exclude the reference fault block. However, these polygons may not be rectangular and we cannot apply Algorithm 2 directly. Each fault-free polygon is partitioned into rectangular regions by the vertical lines that go through the *corners* [8] of the polygon (see Figure 7(c)). A corner is defined as a node on the edge of the polygon and all four neighbors are also in the polygon. In general, k corners divide the polygon into $k+1$ regions. In Figure 7(c), 3 corners divide a fault-free polygon into 4 regions.

In a mesh with several fault blocks, the partition starts from the left-most fault block. Then the recursive procedure is applied to the left and right polygons generated from the partition. These procedures are called in the left-parent-right order (infix order), where each parent is a fault block. As a result, a set of fault-free polygons is generated. We can label these polygons from left to right. In each polygon, we extract the left-most region based on its left-most corner. Then we apply the recursive extraction procedure for the remaining part of this polygon. The regions in each polygon can be labeled from left to right. Algorithm 3 provides such a sequence of regions.

Algorithm 3: Partition in a faulty mesh

MAIN get_polygon (*original_mesh*)

get_polygon (*submesh*) // partition *submesh* into polygons
f = left_most_fault_block (*submesh*)
if $f \neq \phi$
then two_polygon_partition (*submesh*, f, $left\_polygon$, $right\_polygon$)
//partition *submesh* into two polygons.
get_polygon ($left\_polygon$)
get_polygon ($right\_polygon$)
else get_region (*submesh*)

get_region (*polygon*) //partition *polygon* into regions
while (*polygon* $\neq \phi$)
do $left\_most\_region$ = extract (*polygon*, $left\_most\_corner$)
//extract the left-most rectangle
assign_sequence_number ($left\_most\_region$)

For example, in Figure 8(a), block 1 is the left-most fault block. $[0:1,0:12] \cup [2:6,0:1]$ is its left polygon and $[2:6,5:12] \cup [7:9,0:12]$ is its right polygon. It is noted that we use the union of rectangles (regions) to present the area of a polygon. That does not mean we already have these regions before they are partitioned. Applying the recursive procedure for the left polygon, we find it is fault-free and can be partitioned into regions directly. Since it has only one corner $(1,1)$, the left polygon is partitioned into $[0:1,0:12]$ and $[2:6,0:1]$. They are assigned as region-1 and region-2 as the first two regions in the sequence. Applying the recursive procedure to $[2:6,5:12] \cup [7:9,0:12]$ (see Figure 8(b)), we find the left-most fault block $[4:6,9:10]$ (block 2). The left polygon after the partition is $[2:3,5:12] \cup [4:6,5:8] \cup [7:7,0:5]$ and its right polygon is $[4:6,11:12] \cup [7:7,8:12] \cup [8:9,0:12]$. Then, the left-most fault block $[5:7,6:7]$ (block 3) divides the submesh $[2:3,5:12] \cup [4:6,5:8] \cup [7:7,0:5]$ into $[2:3,5:12] \cup [4:4,5:8] \cup [5:6,5:5] \cup [7:7,0:5]$ (left polygon) and $[5:6,8:8]$ (right polygon) (see Figure 8(c)). Based on corners $(3,8)$, $(4,5)$, and $(7,5)$, the left polygon $[2:3,5:12] \cup [4:4,5:8] \cup [5:6,5:5] \cup [7:7,0:5]$ is partitioned into $[2:3,5:12]$, $[4:4,5:8]$, $[5:6,5:5]$, and $[7:7,0:5]$. These four regions are assigned as region-3, region-4, region-5, and region-6, respectively. Since the right polygon $[5:6,8:8]$ is a region, there is no more division and it is assigned as region-7. Finally, the fault-free polygon $[4:6,11:12] \cup [7:7,8:12] \cup [8:9,0:12]$

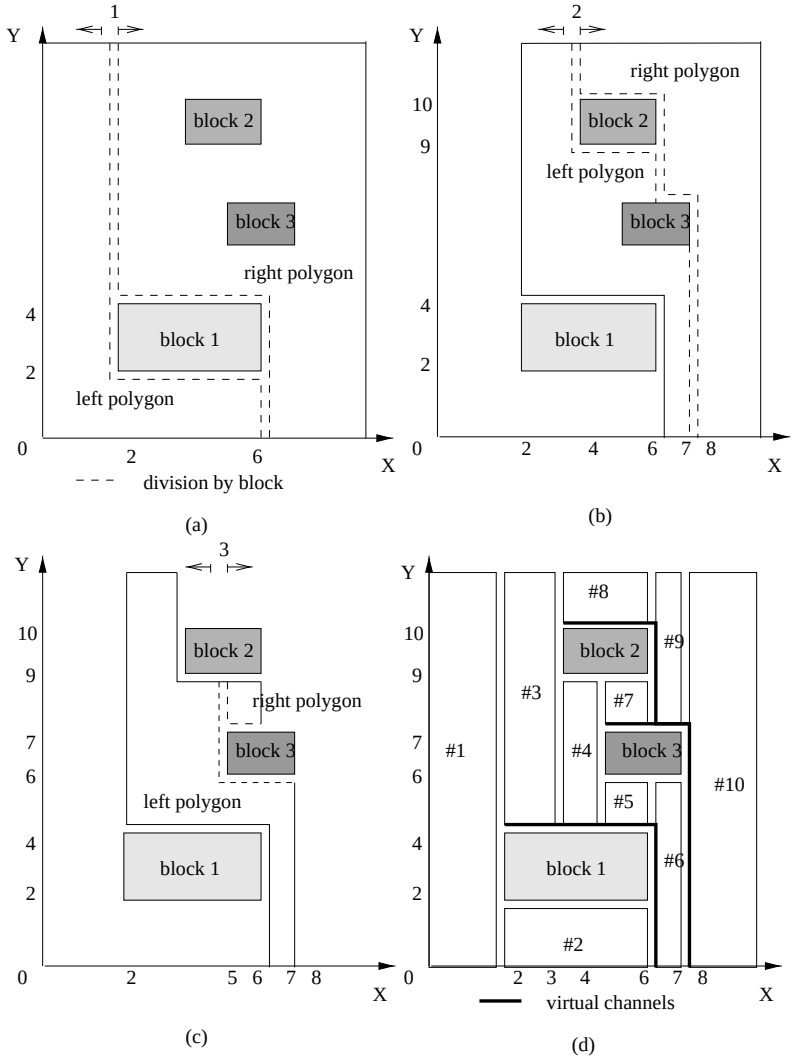

Figure 8: Region division in a mesh with multiple fault blocks.

Algorithm 4: Inter-region-level broadcast

MAIN // from *source* to an eye of each region
e_1: the eye of the same region with *source*
msg_send (*source*, e_1) //source-to-eye unicasting
eye_to_eye_broadcast (e_1, *sequence_of_regions*)

eye_to_eye_broadcast (s, *range*: $[r_s, r_d]$)
//from eye s to an eye in each region in *range*
if $r_s \neq r_d$
then e_2: the eye of the median region in *range*
msg_send (s, e_2)
eye_to_eye_broadcast (s, *half_range_1*: $[r_s, r_{\lfloor \frac{s+d}{2} \rfloor}]$)
eye_to_eye_broadcast (e_2, *half_range_2*: $[r_{\lceil \frac{s+d}{2} \rceil}, r_d]$)
// *half_range_1* + *half_range_2* = *range*

is partitioned into $[4:6,11:12]$, $[7:7,8:12]$, and $[8:9,0:12]$ by corners $(7,11)$ and $(8,8)$ and these regions are assigned accordingly. Now we find a sequence of regions: $[0:1,0:12]$, $[2:6,0:1]$, $[2:3,5:12]$, $[4:4,5:8]$, $[5:6,5:5]$, $[7:7,0:5]$, $[5:6,8:8]$, $[4:6,11:12]$, $[7:7,8:12]$, and $[8:9,0:12]$ (see Figure 8(d)).

Theorem 4: *If the number of fault blocks in an $m \times n$ mesh is f, the number of the regions partitioned by Algorithm 3 is no more than $3f+1$.*

Proof: Assume that an $m \times n$ mesh has only one fault block ($f = 1$). The mesh is partitioned into $4 = 3 + 1$ regions. Assume that the statement is true for $f \leq k$. For the $(k+1)^{th}$ fault block, the fault-free polygon containing it will be partitioned into two fault-free polygons; that is, there is one more fault-free polygon. Each of these two polygons has a new corner caused by the new block. Each new corner incurs a new partition and each new partition will incur a new region. On the other hand, the block will not change the number of regions partitioned by any other corner. That is, the number of the regions partitioned by old corners remains the same. Totally, 3 new regions are generated by the new block. Therefore, (k+1) faults will incur at most $3k+1+3 = 3(k+1)+1$ regions. □

To apply Algorithm 2 in each region, the inter-region-level broadcast to the closest eye of four eyes of each region from a given source is provided in Algorithm 4. The notation of *range* is introduced which spans from the first index to the last index in a sequence of consecutive regions.

Algorithm 5: *msg_send* (s, e_2) $//e_2$ is the closest eye in r_d to s with $r_s < r_d$.

1. Send the passage to the boundary node u along the positive X dimension.
2. If an eligible neighboring region exists, send the message to the closest point (with respect to u) of an eligible neighboring region with the maximum region number.
3. Otherwise, use a virtual channel path to reach the closest point (with respect to u) of r_d.
4. If the current region is not r_d, repeat steps 1 and 2; otherwise, use the X-Y routing algorithm within region r_d to reach e_2.

To broadcast from a source *eye* to the other eyes in the *range*, the *eye* first sends a copy of the message to the closest one of four eyes of the median region in *range*. The *range* is then divided into two subranges by the median value of *range*. The above process is applied individually at these two subranges: one with the source *eye* as its source and the other with an eye in the median region as its source.

For each eye-to-eye unicasting (*msg_send*) from one eye in region-r_s to another eye in region-r_d with $r_s < r_d$, we construct a path of regions: region-r_s $\rightarrow$ region-r_{s+1} $\rightarrow$ ... $\rightarrow$ region-r_{s+k-1} $\rightarrow$ region-r_{s+k} = region-r_d, where r_i $(s \leq i \leq s+k)$ is the sequence number of a region. In addition, the following conditions are satisfied:

1. $r_i < r_{i+1}$ $(s \leq i < s+k)$.
2. region-r_i and region-r_{i+1} $(s \leq i < s+k-1)$ are directly connected.
3. region-r_{s+k-1} and region-r_{s+k} (= region-r_d) are either directly connected or connected through a virtual channel path.

A copy of the message sent by the unicasting will go through all the regions in the sequence. Algorithm 5 shows the procedure of *msg_send*(s, e_2) and the construction of a region sequence from r_s to r_d with $r_s < r_d$. The case for $r_s > r_d$ can be defined in a similar way.

A neighboring region is *eligible* if its region number is within $[r_s, r_d]$. In addition, an eligible neighboring region is connected to the current region directly or via a virtual channel path. *msg_send*(s, e_2) always sends the message to an eligible neighboring region until r_d is reached. If $r_s < r_d$, there are two types of eye-to-eye unicasting, depending on whether a virtual channel path is used or not:

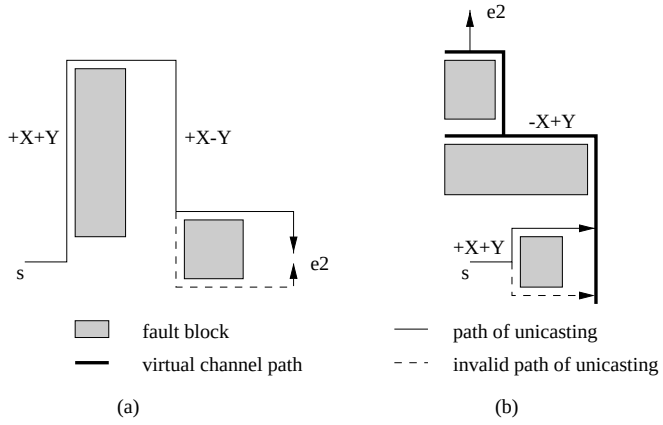

Figure 9: Two types of eye-to-eye unicasting: (a) Type 1 (+X+Y phase followed by +X-Y phase) and (b) type 2 (+X+Y phase followed by -X+Y phase).

1. Type 1 (no virtual channel path is used): a +X+Y phase followed by a +X-Y phase.
2. Type 2 (a virtual channel path is used): a +X+Y phase followed by a -X+Y phase (using a virtual channel path).

Here +X+Y means routing along the positive X and the positive Y drections. Figure 9 shows a graphic illustration of these two types of routing. If several eligible neighboring regions exist, select the one with the maximum region number. In Figure 8(d), if r_s = region-1 and r_d = region-10, a type 1 path: region-1 $\rightarrow$ region-3 $\rightarrow$ region-8 $\rightarrow$ region-9 $\rightarrow$ region-10 can be derived; specifically, phase 1 (+X+Y) consists of region-1 $\rightarrow$ region-3 $\rightarrow$ region-8 and phase 2 (+X-Y) consists of region-8 $\rightarrow$ region-9 $\rightarrow$ region-10. Both region-3 and region-2 are eligible regions of region-1. Based on Algorithm 5, region-3 is selected as the successive region. Then, by the same reason, region-8 is selected as the successive region. After that, region-9 is selected since it is the only eligible region of region-8. Finally, region-10 is selected and the destination region r_d is reached. If r_s = region-2 and r_d = region-8, a type 2 path: region-2 $\rightarrow$ region-6 $\rightarrow$ region-8 can be derived. For region-2, region-3, region-4, and region-5 are eligible regions connected by a virtual channel path and region-6 is the only eligible region connected directly. Thus, region-6 with the maximum region number is selected. After that, region-8 is selected because region-8 (r_d) can be connected to region-6 through a virtual channel path and it is the only eligible region of region-6. In this case, phase 1 (+X+Y) is region-2 $\rightarrow$ region-6 and phase 2 (-X+Y) using a virtual channel path is region-6 $\rightarrow$ region-8.

When $r_s > r_d$, again two types of unicasting exist:

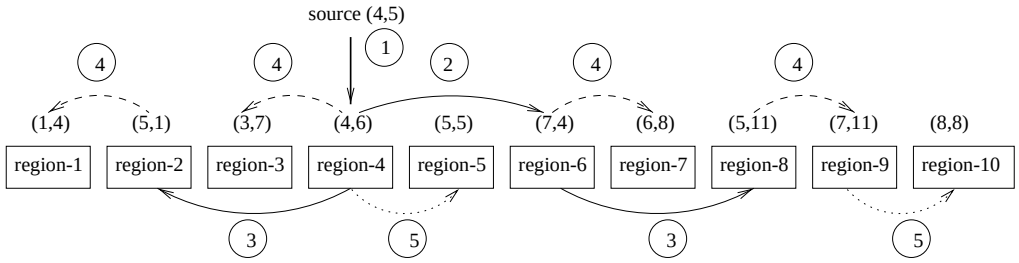

Figure 10: Inter-region-level broadcast for the example shown in Figure 8(d).

1. Type 1 (no virtual channel path is used): a -X-Y phase followed by a -X+Y phase.
2. Type 2 (a virtual channel path is used): a +X-Y phase (using a virtual channel path) followed by a -X+Y phase.

In Figure 8, (r_s, r_d) = (region-10, region-1) is a type 1 unicasting and the corresponding path is: region-10 $\rightarrow$ region-6 $\rightarrow$ region-2 $\rightarrow$ region-1. (r_s, r_d) = (region-8, region-2) is a type 2 unicasting and the corresponding path is region-8 $\rightarrow$ region-6 $\rightarrow$ region-2, where region-8 $\rightarrow$ region-6 uses a virtual channel path. The virtual channel path is used at most once at the first step and the region with the minimum region number is always selected if there are several eligible regions.

Note that the distance of routing path in an eye-to-eye unicasting has an upper bound although the number of regions used may not be the least. Such a path only uses the channels inside the regions within the *range*. The virtual channel path of a block is used only if two consecutive regions disconnected by the block are both inside the *range*. In addition, different *range*s at the same broadcast step have no overlap. Thus, any eye-to-eye unicasting does not interfere with communication outside *range*. This is a key for contention-free broadcast which ensures a deadlock-free broadcast. If the number of fault blocks is no more than f, the inter-region-level broadcast will complete in $\lceil \lg(3f+1) \rceil$ steps since the number of regions is no more than $3f+1$ (based on Theorem 4). When the source is not an eye, one extra step is needed to send the message to the closest eye in the same region.

For the mesh shown in Figure 8(d), there are ten regions. Assume that $(4,5)$ is the source. First, it sends a message to the closest eye $(4,6)$ in the same region (region-4). After that, the message received at $(4,6)$ will be sent to an eye of each region through the inter-region-level broadcast. Since region-6 is the median one among these ten regions, $(4,6)$ will send a copy of the message to the closest eye of region-6 in the second step. Since region-5 is the only eligible region of region-4,

step	1	2	3		4				5	
source region	4	4	4	6	2	4	6	8	4	9
destination region	4	6	2	8	1	3	7	9	5	10
path	$4 \rightarrow 4$	$4 \rightarrow 5 \rightarrow 6$	$4 \Rightarrow 2$	$6 \Rightarrow 8$	$2 \rightarrow 1$	$4 \rightarrow 3$	$6 \Rightarrow 7$	$8 \rightarrow 9$	$4 \rightarrow 5$	$9 \rightarrow 10$

Table 1: List of region paths for the example shown in Figure 8(d): directly connected path ($\rightarrow$) and connected path through virtual channel path ($\Rightarrow$).

Algorithm 6: Fault-tolerant broadcast algorithm

- Build fault blocks and divide the $m \times n$ mesh into a sequence of regions (Algorithm 3).
- Calculate the positions of eyes in each region.
- If there is more than one region, inter-region-level broadcast (Algorithm 4) is applied to send a copy of the message to one eye in each region.
- Intra-region-level broadcast (Algorithm 2) is applied in each region.

---

the message passes through $(5, 5)$ which is the closest node in region-5 from $(4, 6)$. Then, by the same reason, the message passes through $(7, 5)$ and arrives at the closest eye $(7, 4)$ of region-6. For the first five regions, region-2 is the median region and $(4, 6)$ sends a copy of the message to the closest eye of region-2 in the third step. Because region-2 is disconnected with region-4, the virtual channel path between $(4, 5)$ and $(7, 1)$ is used to forward the message to $(5, 1)$. At the same time, $(7, 4)$ sends a copy of the message to the median region (region-8) of the other five regions. Since region-7 and region-8 can be connected to region-6 through a virtual channel path and they are all eligible regions of region-6, region-8 with the maximum region number is selected and the virtual channel path between $(8, 4)$ and $(5, 11)$ is used to forward the message to $(5, 11)$ and avoid the access of region-7. In the fourth step, the eyes $(5, 1)$, $(4, 6)$, $(7, 4)$, and $(5, 11)$ of regions 2, 4, 6, 8 send copies of the message to the closest eyes $(1, 4)$, $(3, 7)$, $(6, 8)$, and $(7, 11)$ of regions 1, 3, 7, and 9, respectively. At the last step, the eyes $(4, 6)$ in region-4 and $(7, 11)$ in region-9 send messages to the closest eyes $(5, 5)$ of region-5 and $(8, 8)$ of region-10 and complete this inter-region-level broadcast. All the steps are shown in Figure 10 and the path of each step is shown in Table 1.

The major steps of the proposal fault-tolerant broadcasting are listed in Algorithm 6.

Theorem 5: *The broadcasting in Algorithm 6 has no traffic contention.*

Proof : We only need to prove that Algorithm 6 has no step contention, assuming that the broadcast message is relatively short; that is, broadcasting within a step completes quickly so that its contention with broadcasting in the next step is negligible. For the inter-region-level broadcast, different *range*s have no overlap at the same step and there is only one eye-to-eye unicasting in each *range*. Algorithm 4 ensures that each unicasting in a step has an independent path (see Figure 10). According to the construction of the path, such an eye-to-eye unicasting only uses the channels inside the regions within the *range*. Since a virtual channel path is used only if two consecutive regions disconnected by a block are both in the *range*, each virtual channel path is used at most once at each step. Thus, inter-region-level broadcast is contention-free. The intra-region-level broadcast is a simple version of inter-region-level broadcast. Since each region is fault-free, no virtual channel path is needed. Therefore, these is no step contention. (A formal proof of contention-free for Algorithm 1 [1] can be easily adopted here for Algorithm 2 by replacing each square by a rectangle.) □

Last, an upper bound of TCD in an $m \times n$ mesh with f fault blocks is given.

Theorem 6: *In an $m \times n$ mesh with f fault blocks, Algorithm 6 completes a broadcast within $1 + \lceil \lg(3f+1) \rceil + \lceil \lg m \rceil + \lceil \lg n \rceil$ steps and its TCD is no more than $(3f+1)(2m+2n+ED_{m\times n} - m \times n) + m \times n + 3f$, where $ED_{m\times n}$ is the TCD of an $m \times n$ fault-free mesh.*

5 Conclusions

We have provided a broadcast in an $m \times n$ rectangular mesh with randomly generated faults and studied its upper bound of total communication distance (TCD). The mesh is partitioned into a set of fault-free rectangular regions based on the locations of fault blocks. A fault-tolerant broadcast is carried out at two levels: inter-region and intra-region. In the inter-region-level broadcast, the broadcast message is sent from a given source to an eye in each region. Then in the intra-region-level broadcast, the extended optimal fault-free broadcast is applied from the selected eye within each region. Applying this approach to other topologies is one direction of our future research.

References

[1] S. Cang and J. Wu. Minmizing total communication distance of a time-step optimal broadcast in mesh network. *Proc. of the First Merged IPPS/SPDP 1998*, 10-17, April, 1998.

[2] S. L. Johnsson and C.T. Ho. Optimal broadcasting and personalized communication in hypercubes. *IEEE Trans. Computer*, 38(9), 1249-1268, Sept., 1989.

[3] S. L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problems", *ACM Trans. Progr. Languages Systems*, June, 1992, 633-639.

[4] D. K. Panda. Issues in designing efficient and practical algorithms for collective communication on wormhole routed system. *Proc. of the 1995 ICPP Workshop on Challenges for Parallel Processing*, 8-15, Aug., 1995.

[5] J. Y. L. Park, H. A. Choi, N. Nupairoj, and L. M. Ni. Construction of optimal multicast trees based on the parameterized communication model. *Proc. of the International Conference on Parallel Processing*, 180-187, Aug., 1996.

[6] P. Ramanathan, K. G. Shin and R. W. Butler. "Fault-tolerant clock synchronization in distributed systems" *Computer*, 23, Oct., 1990, 33-42.

[7] J. Wu. Fault-Tolerant Adaptive and Minimal Routing in Mesh-Connected Multicomputers Using Extended Safety Levels. *IEEE Trans. Parallel and Distributed Systems*, 11(2), 149-159, Feb., 2000.

[8] J. Wu. A Distributed Formation of Orthogonal Convex Polygons in Mesh-Connected Multicomputers. *Proc. of International Parallel and Distributed Processing Symposium (IPDPS)*, April, 2001.

[9] Z. Jiang and J. Wu. Fault-Tolerant Broadcasting in 2-D Wormhole-Routed Meshes. Department Technical Reports, TR-CSE-01-09, Florida Atlantic University, April, 2001.

Appendix

Lemma 2: *The TCD of the extended optimal broadcasting in an $m \times n$ fault-free mesh, $ED_{m\times n}$, can be calculated by*

$$ED_{m\times n} = ED'_{m\times n} + m \times n - 1$$

where $ED'_{m\times n} = A'_m + ED'_{\lceil \frac{m}{2} \rceil \times n} + ED'_{\lfloor \frac{m}{2} \rfloor \times n}$ and $A'_m = D_{\lceil \frac{m}{2} \rceil} + D_{\lfloor \frac{m}{2} \rfloor} = A_m - 1$. Moreover, $ED'_{m\times n} \geq ED'_{m'\times n'}$ if $m \geq m' \geq 1$ and $n \geq n' \geq 1$.

Proof: For any eye-to-eye unicasting within the intra-region-level broadcast in an $m \times n$ fault-free mesh, any node u (except the source node) will receive a copy of the message only once and it comes from an upper level eye e which is at least one hop from u. The remaining distance of such a transmission can be defined as $A'_k = A_k - 1 \geq 0$, where A_k is the distance from e to u. Therefore, the TCD can be defined as the sum of distance of those $m \times n - 1$ hops and all the remaining hops: $ED_{m\times n} = m \times n - 1 + ED'_{m\times n}$, where $ED'_{m\times n} = A'_m + ED'_{\lceil \frac{m}{2} \rceil \times n} + ED'_{\lfloor \frac{m}{2} \rfloor \times n}$. Based on Lemma 1, $A'_m \geq A'_{m'}$ if $m \geq m' \geq 1$. Thus, it is easy to derive by induction that $ED'_{m\times n} \geq ED'_{m'\times n'}$ if $m \geq m' \geq 1$ and $n \geq n' \geq 1$. (see the proof of Theorem 3). □

Lemma 3: *In an $m \times n$ mesh with fault blocks, the distance of an eye-to-eye unicasting from region-r_s to region-r_d is no more than $2m + 2n$.*

Proof: Case 1 for ($r_s < r_d$): If the eye-to-eye unicasting is type 1, phase 1 is a +X+Y routing and phase 2 is a +X-Y routing. Clearly, there is no detour along the X dimension and the distance along the X dimension is bounded by m. +Y routing in phase 2 followed by -Y routing (in phase 2) will generate at most $2n$ hops. Therefore, overall distance is bounded by $m + 2n$. If the eye-to-eye unicasting is type 2, phase 1 is a +X+Y routing (like phase 1 of type 1 routing) and phase 2 is a -X+Y routing using a virtual channel path. Clearly, there is no detour along the Y dimension and the distance along the Y dimension is bounded by n. +X routing in phase 1 followed by -X routing (in phase 2) will generate at most $2m$ hops. Therefore, overall distance is bounded by $2m + n$. Case 2 for ($r_s > r_d$): The type 1 eye-to-eye unicasting consists of a -X-Y phase followed by a -X+Y phase. Overall distance is bounded by $m + 2n$. The type 2 eye-to-eye unicasting consists of a +X-Y phase followed by a -X+Y phase. Overall distance is bounded by $2m + 2n$. Combining the above two cases, we have the bounded value $2m + 2n$. □

Proof of Theorem 6

For the first step of the inter-region level broadcast, a given source has an optimal path to the eye of the same region. Its distance is no more than $m + n$ ($< 2m + 2n$). Based on Theorem 4, there are at most k ($\leq 3f + 1$) regions if there are f fault blocks. It needs at most $\lceil \lg k \rceil$ steps and totally there are $k - 1$ eye-to-eye unicasting. For each eye-to-eye unicasting, the distance is no more than $2m + 2n$ (based on Lemma 3). Totally, it needs at most $1 + \lceil \lg k \rceil$ ($\leq 1 + \lceil \lg(3f + 1) \rceil$) steps and its TCD is no more than $k(2m + 2n)$ ($\leq (3f + 1)(2m + 2n)$).

Since each region is no bigger than the whole mesh, the intra-region-level broadcast in an $m' \times n'$ region completes in at most $\lceil \lg m \rceil + \lceil \lg n \rceil$ steps and based on Lemma 2 its TCD ($ED_{m' \times n'} = ED'_{m' \times n'} + m' \times n' - 1$) is no more than $ED'_{m \times n} + m' \times n' - 1$. Thus, the TCD of the intra-region-level broadcast in the whole mesh is $\sum_{i=1}^{k} ED_{m_i \times n_i}$, where $ED_{m_i \times n_i}$ is the TCD of region-i (an $m_i \times n_i$ rectangle). Such a TCD is no more than $\sum_{i=1}^{k} (ED'_{m \times n} + m_i \times n_i - 1) \leq k * ED'_{m \times n} + m \times n - 1 = k * (ED_{m \times n} - m \times n + 1) + m \times n - 1 \leq (3f + 1)(ED_{m \times n} - m \times n) + m \times n + 3f$. Therefore, the fault-tolerant broadcast completes within $1 + \lceil \lg(3f + 1) \rceil + \lceil \lg m \rceil + \lceil \lg n \rceil$ steps and its TCD is no more than $(3f + 1)(2m + 2n + ED_{m \times n} - m \times n) + m \times n + 3f$. □