# Minimizing Total Communication Distance of a Time-Step Optimal Broadcast in Mesh Networks

Songluan Cang and Jie Wu
Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431
{scang, jie}@cse.fau.edu

## Abstract

*In this paper, we propose a new minimum total communication distance ($TCD$) algorithm and an optimal $TCD$ algorithm for broadcast in a 2-dimension mesh. The former generates a minimum $TCD$ from a given source node, and the latter guarantees a minimum $TCD$ among all the possible source nodes. These algorithms can be generalized to a $d$-dimensional mesh or torus. $TCD$ can potentially be used as a measurement for other types of collective communication operations.*

**Keywords**: Broadcast, communication distance, mesh, torus.

## 1. Introduction

In a multicomputer system, a collection of processors (also called nodes) work together to solve large application problems. The mesh-connected topology is one of the most thoroughly investigated network topologies. It is of large importance due to its simple structure and its good performance in practice and is becoming popular for reliable and high-speed communication switching.

In order to minimize communication latency it is importance to design an efficient implementation of collective communication operations [5] which include multicast and broadcast. Multicast is an important system-level communication service [4] in which the same message is delivered from a source to an arbitrary number of destination nodes. Broadcast [3] is a special case of multicast in which the same message is delivered to all the nodes.

A major source of communication delay for broadcast in a network is the communication time spent on sending messages from one node to all the other nodes. This communication time is influenced by many factors. One important factor is the traffic generated during the broadcast process. We measure such traffic by a *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 $TCD$ has become an important issue in designing an efficient broadcast. A *minimum* $TCD$ algorithm for broadcast in a mesh starting from a given source node is the one that generates the minimum $TCD$ among all the possible $TCD$s from the same source node. An *optimal* $TCD$ algorithm is the one that generates a minimum $TCD$ among $TCD$s for all the possible source nodes, not just for a given source node.

Given a 2-dimensional (2-D) mesh, say an $n \times n$ mesh with $n = 2^k$, where $k$ is a non-negative integer ($k$ is used as an integer throughout this paper), we only consider broadcast algorithms that can complete a broadcast in $\log n^2 = 2k$ time steps, i.e., a time-step optimal broadcast algorithm will be simply denoted as a broadcast algorithm. Under the cut-through switching technique [2], 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. We assume that in each time step a node may do one of the following: sending a message to one node, receiving a message from one node, or being idle. The challenge in designing a minimum $TCD$ of a time-step optimal broadcast algorithm (for a given source node) is to generate a routing path that guarantees a minimum $TCD$ without traffic contention at any time step.

One related work [6] deals with the simplest case in which the source node of broadcast is always a corner node of a given 2-D mesh. This case rarely occurs in real applications. Therefore, we need to find a minimum $TCD$ broadcast algorithm for any given source node. In this paper, we identify certain unique nodes called *eyes*. If we start a broadcast from one of these eyes and follow certain rules defined in this paper, we will obtain an optimal $TCD$.

Specifically, we propose: (1) A minimum $TCD$ algorithm for a given source node in a 2-D mesh. (2) An optimal $TCD$ algorithm for a 2-D mesh. (3) Expressions of optimal $TCD$ and minimum $TCD$ for a given source node. (4) Extensions of the above results to a $d$-D mesh and a $d$-D torus.

The remainder of the paper is organized as follows. Section 2 shows some simple examples and describes the notation used in the paper. In Section 3, we provide our major results on $TCD$s for 2-D meshes. We generalize our results to $d$-D meshes in Section 4. In Section 5, we conclude this paper.

## 2. Notation and Examples

For a given $n \times n$ mesh with $n = 2^k$, we assume that the distance between any two adjacent nodes is one. The location of a node in a mesh is denoted by a pair of coordinates $(x, y)$. The origin of the coordinate system is assumed to be the upper-left corner of the mesh, as shown in Fig.1 (a). Both $x$ and $y$ are integers $(x, y = 0, 1, 2, ..., n-1)$. The node at $(x, y)$ is denoted by $N(x, y)$.

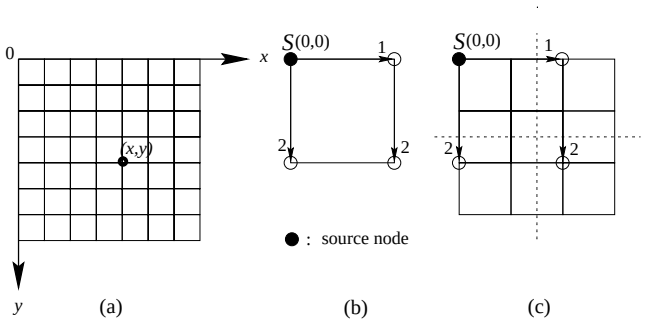


**Figure 1. (a) A mesh with its coordinate system. (b) Broadcast in a $2 \times 2$ mesh. (c) Broadcast in a $4 \times 4$ mesh.**

**Definition 1**: *Denote $D_k(x, y)$ as the $TCD$ of a broadcast algorithm originated from a source node $S(x, y)$ in a $2^k \times 2^k$ mesh and $MD_k(x, y)$ as the minimum $TCD$ originated from a source node $S(x, y)$ among all the possible broadcast algorithms. Obviously, $MD_k(x, y) = min\{D_k(x, y)\}$.*

Denote $FD$ as the communication distance in the first step of a broadcast, $SD$ as the communication distance in the second step, and $RD$ as the communication distance in the remaining steps. Obviously, for a given source node $S(x, y)$ in a mesh, different algorithms lead to different sets of $FD$, $SD$ and $RD$. Clearly, $D_k(x, y) = FD + SD + RD$ for a particular broadcast algorithm.

Now let's look at some simple broadcast examples with the source node being the upper-left corner node of a given mesh. A $2 \times 2$ mesh $(k = 1)$ is the simplest case. Fig.1 (b) shows the process of a broadcast starting from node $S(0, 0)$. Arrows 1 and 2 represent the first and second steps of broadcast, respectively. The $TCD$ of this case is calculated by $D_1(0, 0) = FD + SD = 1 + 2 = 3$. Obviously, starting from any node in this mesh, we will obtain the same result. This means that $MD_1(x, y) = D_1(x, y) = 3$, where $x$, $y$ = 0 or 1. A $2 \times 2$ mesh is a basic unit for broadcast and is called a *unit mesh*.

In a time-step optimal broadcast, after each step the number of nodes having received the message must be doubled. As shown in Fig.1 (c), we divide the given $4 \times 4$ mesh into four $2 \times 2$ submeshes. By treating each of these submeshes as a *virtual node*, we reduce a $4 \times 4$ mesh problem to a $2 \times 2$ mesh one. Therefore, in the first two steps we can use the same approach for a $2 \times 2$ mesh with $FD = 2$ and $SD = 4$. After two steps, all the virtual nodes receive the message. In the remaining two steps, each of these four $2 \times 2$ submeshes completes the broadcast process within its submesh with $D_1(0, 0) = 3$, i.e., $RD = 4 \times 3 = 12$. Overall, $D_2(0, 0) = FD + SD + RD = 2 + 4 + 12 = 18$. This turns out to be the minimum $TCD$ by comparing it with results of all the other arrangements; that is, $MD_2(0, 0) = 18$. It is not difficult to derive $MD_2(0, 1) = 16$ and $MD_2(1, 1) = 15$ in the same way.

However, in a $8 \times 8$ mesh, the minimum $TCD$ cannot be easily determined as in the above two examples. If the source sends the message to the upper-left corner nodes of the other three $4 \times 4$ submeshes, i.e., $N(4, 0)$, $N(0, 4)$, and $N(4, 4)$, in the first two steps, we have $D_3(0, 0) = 84$, which is not $MD_3(0, 0)$, because there are better results. For examples, if $N(5, 0)$ is the destination node of the first step, $N(0, 5)$, and $N(5, 4)$ are the destination nodes of the second step, $D_3(0, 0) = 80$. If $N(5, 1)$ is the destination node of the first step, $N(1, 5)$ and $N(5, 5)$ are the destination nodes of the second step, $D_3(0, 0) = 79$. Therefore, to find the minimum $TCD$, the first two steps are extremely important. The selection of the upper-left corner node of the upper-right submesh in the first step may not generate a minimum result. The exact location depends on the size of the given mesh as we will discuss in the next section.

To give some insights on the location(s) of the source node that can generate an optimal $TCD$ of a given $2^k \times 2^k$ mesh, we calculate the minimum $TCD$ for each node in the mesh. Fig.2 (a), (b), and (c) show the minimum $TCD$ for each node in a $4 \times 4$, $8 \times 8$, and $16 \times 16$ mesh, respectively. We place results for each mesh in a matrix. Within such a matrix, the number at a particular position represents the minimum $TCD$ of the node corresponding to this position

in the corresponding mesh.

Clearly, each mesh has an optimal $TCD$, which appears at four different locations (marked with an underline) in the matrix. These locations are in four different submeshes and are called eyes of the mesh to be defined in the next section. For the $16 \times 16$ mesh, we just show the upper-left $8 \times 8$ submatrix, since the matrix is symmetric with respect to the center of the matrix.

$$\begin{pmatrix} 18 & 17 & 16 & 18 \\ 16 & \underline{15} & \underline{15} & 17 \\ 17 & \underline{15} & \underline{15} & 16 \\ 18 & 16 & 17 & 18 \end{pmatrix}$$

$(a)$

$$\begin{pmatrix} 79 & 76 & 74 & 76 & 74 & 74 & 75 & 79 \\ 75 & 72 & 71 & 73 & 70 & 70 & 72 & 76 \\ 74 & 70 & \underline{69} & 70 & 70 & \underline{69} & 71 & 74 \\ 74 & 70 & 70 & 71 & 71 & 70 & 73 & 76 \\ 76 & 73 & 70 & 71 & 71 & 70 & 70 & 74 \\ 74 & 71 & \underline{69} & 70 & 70 & \underline{69} & 70 & 74 \\ 76 & 72 & 70 & 70 & 73 & 71 & 72 & 75 \\ 79 & 75 & 74 & 74 & 76 & 74 & 76 & 79 \end{pmatrix}$$

$(b)$

$$\begin{pmatrix} 318 & 313 & 309 & 311 & 307 & 306 & 307 & 311 \\ 312 & 307 & 304 & 306 & 301 & 300 & 302 & 306 \\ 309 & 303 & 300 & 301 & 299 & 297 & 299 & 302 \\ 307 & 301 & 299 & 300 & 298 & 296 & 299 & 302 \\ 307 & 302 & 297 & 298 & 296 & 294 & 294 & 298 \\ 303 & 298 & 294 & 295 & 293 & \underline{291} & 292 & 296 \\ 304 & 298 & 294 & 294 & 295 & 292 & 293 & 296 \\ 307 & 301 & 298 & 298 & 298 & 295 & 297 & 300 \end{pmatrix}$$

$(c)$

**Figure 2. The minimum $TCD$s for each node in (a) a $4 \times 4$ mesh, (b) a $8 \times 8$ mesh, and (c) a $16 \times 16$ mesh.**

## 3. Minimizing $TCD$ of a Time-Step Optimal Broadcast in a 2-D Mesh

In the following definition, we define special nodes, called eyes, in a given mesh. We will show later that the minimum $TCD$ with respect to an eye is the optimal $TCD$.

**Definition 2**: *There are four* eyes *in a $2^k \times 2^k$ mesh, labeled as $E_k(i)$, $i = 1, 2, 3, 4$. These eyes are recursively defined as follows: All four nodes in a $2 \times 2$ mesh are eyes, $E_1(i)$, $i = 1, 2, 3, 4$, as shown in Fig.3 (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, $E_{k-1}(i)$. Eyes, $E_k(i)$, $i = 1, 2, 3, 4$, are selected from sixteen $E_{k-1}(i)$s. Specifically, eyes $E_k(i)$ are the four $E_{k-1}(i)$s that are the closest to the center of the $2^k \times 2^k$ mesh, as shown in Fig.3 (c).*

For examples, the inner four nodes of a $4 \times 4$ mesh, as shown in Fig.3 (b), are eyes, $E_2(i)$, $i = 1, 2, 3, 4$. Sometimes, we also use $E_k$ to represent $E_k(i)$ to simplify our notation, and denote $i = 1, 2, 3, 4$ as the index of the upper-left, upper-right, lower-left, and lower-right submesh and eye, respectively, and we follow this convention throughout this paper.

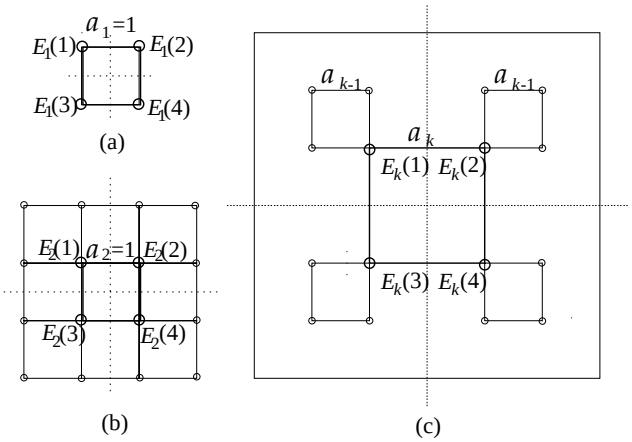


**Figure 3. The recursive definition of eyes of (a) a $2 \times 2$ mesh, (b) a $4 \times 4$ mesh, and (c) a $2^k \times 2^k$ mesh.**

**Definition 3**: *Define the square, formed by four eyes $E_k(i)$ of a $2^k \times 2^k$ mesh as its four corners, to be the* eye-square *of the $2^k \times 2^k$ mesh. Denote $a_k$ as the length of the side of this eye-square.*

Based on Definition 2, it is obvious from Fig.3 (c) that two $a_k$s plus two $a_{k-1}$s equals the length of side of the $2^k \times 2^k$ mesh plus one, i.e., $2a_k + 2a_{k-1} = n = 2^k$, where $a_{k-1}$ is the length of the side of eye-square of a $2^{k-1} \times 2^{k-1}$ mesh. This immediately leads to

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

and $a_1 = 1$. This recursive formula leads to

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

Using Eq. (2), we can easily determine locations of all four eyes of a given $2^k \times 2^k$ mesh.

By taking the advantage of the recursive definition of eyes, we propose the following broadcast algorithm: If we start a broadcast from eye $E_k(1)$, it first sends the message to $E_k(2)$, then these two eyes send the message to $E_k(3)$ and $E_k(4)$, respectively. Each of four $2^{k-1} \times 2^{k-1}$ submeshes will have one node with the message after the above two steps. According to the definition of eyes, each of these four nodes is also an eye of the submesh. Repeating the

above procedure, i.e., each of the four submeshes delivers the message to eyes within its submesh, the message will be delivered down to submeshes level by level, and the broadcast completes when all the nodes obtain the message. Note that $D_k(E_k(i))$s ( $i = 1, 2, 3, 4$ ) are the same, due to the symmetry of the mesh. Therefore, we can use $D_k(E_k)$ to represent them to simplify the notation. Clearly, $D_k(E_k)$ can be calculated recursively by the following formula

$$D_k(E_k) = 3a_k + 4D_{k-1}(E_{k-1}). \quad (3)$$

We will show later in Theorem 2 that this $D_k(E_k)$ corresponds to the optimal $TCD$. Now we consider an algorithm in which a broadcast starts from any node $S$ in the mesh, but the message are still forwarded to the eyes of the mesh in the first two steps.

**Algorithm 1**: (Minimum $TCD$ broadcast algorithm for a given source node $S$ in a $2^k \times 2^k$ mesh.)

- 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 the source node $S$ is in the upper-left submesh, as shown in Fig.4.
- The source node sends the message to the upper-right eye $E_k(2)$ in the first step.
- In the second step, the source node sends the message to either the lower-left eye $E_k(3)$ or $E_{k-1}(1)$ depending on which one is closer to the source node, and $E_k(2)$ sends the message to the lower-right eye $E_k(4)$, as shown in Fig.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 submeshes level by level until reaching the unit meshes, $2 \times 2$ meshes, and all these unit meshes complete the broadcast within themselves in two steps.

Based on the definition of eyes, each node in a given mesh is an eye of exactly one submesh (including the given mesh). Each eye will be visited exactly once in Algorithm 1.

To calculate $D_k(S)$, the $TCD$ obtained from Algorithm 1, we use a new set of relative coordinate systems. For a $2^k \times 2^k$ mesh, we set up a $u_k - v_k$ coordinate system with the origin being on $E_k(1)$, as shown in Fig.4. Each of the four $2^{k-1} \times 2^{k-1}$ submeshes has its own coordinate system. For example, the $u_{k-1}(i) - v_{k-1}(i)$ coordinate system is for the $i$th submesh (see Fig.5).

Now assume that $S$ is $u_k$ and $v_k$ away from $E_k(1)$ in the $u_k$ and $v_k$ direction respectively, as shown in Fig.4, i.e., the coordinates of $S$ are $(u_k, v_k)$. Also $S$ is in the upper-left submesh, because we can always do so by rotating the mesh.

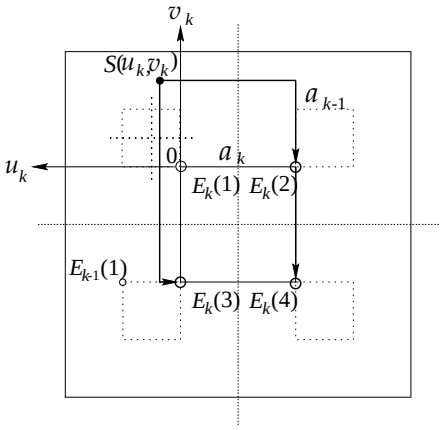


**Figure 4. The $u_k - v_k$ coordinate system in a $2^k \times 2^k$ mesh and calculation of the first two steps of $D_k(S)$.**

The expression for $D_k(S)$ varies based on different locations of $S(u_k, v_k)$ in the upper-left submesh. For examples, if $0 < u_k \leq a_{k-1}/2$, and $0 < v_k \leq 2^{k-1} - 1 - a_k/2$, as shown in Fig.4 and region $(iv)$ in Fig.6, $FD = a_k + u_k + v_k$, $SD = (a_k + u_k + v_k) + a_k$, and $RD = D_{k-1}(S) + 3D_{k-1}(E_{k-1})$. So the $D_k(S)$ in this case is

$$D_k(S) = 2u_k + 2v_k + 3a_k + D_{k-1}(S) + 3D_{k-1}(E_{k-1}).$$

If $-a_k/2 < u_k \leq 0$ and $0 < v_k \leq 2^{k-1} - 1 - a_k/2$, the region $(ii)$ in Fig.6, $FD = a_k + u_k + v_k$, $SD = (a_k - u_k + v_k) + a_k$, and $RD = D_{k-1}(S) + 3D_{k-1}(E_{k-1})$. So the $D_k(S)$ in this case is

$$D_k(S) = 2v_k + 3a_k + D_{k-1}(S) + 3D_{k-1}(E_{k-1}).$$

There are totally six different expressions for $D_k(S)$ corresponding to six different locations of $S$. To simplify our discussion, we introduce a function $f_k(u_k, v_k)$ (or simply $f_k(S)$) so that we can represent $D_k(S)$ with a general expression.

**Lemma 1**: $D_k(S)$ *can always be expressed by a general form:*

$$D_k(S) = f_k(u_k, v_k) + [3a_k + D_{k-1}(S) + 3D_{k-1}(E_{k-1})], \quad (4)$$

*where* $k \geq 2$, *and*

$$f_k(u_k, v_k) = \begin{cases} 0 & (i) \\ 2v_k & (ii) \\ 2u_k & (iii) \\ 2u_k + 2v_k & (iv) \\ u_k + |u_k - a_{k-1}| & (v) \\ u_k + |u_k - a_{k-1}| + 2v_k & (vi) \end{cases} \quad (5)$$

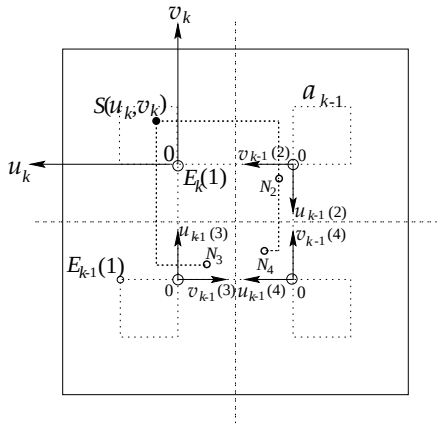


**Figure 5. Calculation of $D'_k(S)$ in a $2^k \times 2^k$ mesh.**

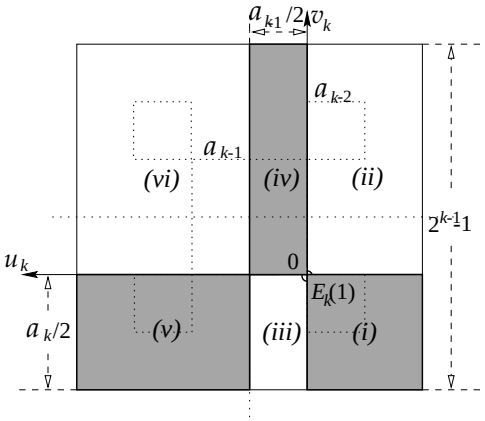


**Figure 6. Six different regions of the source node $S(u_k, v_k)$ in the upper-left submesh of a $2^k \times 2^k$ mesh.**

*where $(i)$ to $(vi)$ represent six regions in the upper-left submesh where the source node locates (see Fig.6):*

$$
\begin{array}{ll}
(i): & (-a_k/2 < u_k \le 0) \wedge (-a_k/2 < v_k \le 0) \\
(ii): & (-a_k/2 < u_k \le 0) \wedge (0 < v_k \le 2^{k-1} - 1 - a_k/2) \\
(iii): & (0 < u_k \le a_{k-1}/2) \wedge (-a_k/2 < v_k \le 0) \\
(iv): & (0 < u_k \le a_{k-1}/2) \wedge (0 < v_k \le 2^{k-1} - 1 - a_k/2) \\
(v): & (a_{k-1}/2 < u_k \le 2^{k-1} - 1 - a_k/2) \wedge \\
 & (-a_k/2 < v_k \le 0) \\
(vi): & (a_{k-1}/2 < u_k \le 2^{k-1} - 1 - a_k/2) \wedge \\
 & (0 < v_k \le 2^{k-1} - 1 - a_k/2)
\end{array}
$$

In the second step of Algorithm 1, if the source node is in $(v)$ or $(vi)$, it sends the message to $E_{k-1}(1)$; otherwise, it sends to $E_k(3)$.

Function $f_k(S)$ is a function of the location of the source node $S$. It is always greater than or equal to zero, i.e.,

$$f_k(u_k, v_k) \ge 0. \tag{6}$$

By comparing Eq. (3) and (4), it is easy to see that

$$D_k(S) - D_k(E_k) = f_k(S) + D_{k-1}(S) - D_{k-1}(E_{k-1}).$$

Repeatly substituting $D_{k-1}(S)$ and $D_{k-1}(E_{k-1})$ in the above equation by Eq. (3) and (4), respectively, we have

$$D_k(S) = \sum_{i=2}^{k} f_i(S) + D_k(E_k). \tag{7}$$

Because each $f_i(S) \ge 0$ in Eq. (7) is greater than 0, we immediately have

$$D_k(S) \ge D_k(E_k),$$

where the equal sign is taken only when $S = E_k$. This result leads to the following theorem.

**Theorem 1**: *If the source node is an eye of the mesh, the TCD obtained by Algorithm 1 is the minimum among all the possible source nodes that also use Algorithm 1.*

The following theorem shows that the Algorithm 1 is the best possible broadcast algorithm.

**Theorem 2**: *The $TCD$ obtained from Algorithm 1 is the minimum $TCD$ for a given source node.*

The proof of Theorem 2 is in Appendix. One special application of the minimum $TCD$ algorithm is when the source node is a corner node of a given mesh. In this case, the source node doesn't need to compare $E_k(3)$ and $E_{k-1}(1)$ in the second step of the algorithm. It just sends the message to $E_{k-1}(1)$. Clearly, our result here is a generalization of the one in [6], where the source node is restricted to a corner node.

In fact, the minimum $TCD$ for $S(0,0)$, the upper-left corner node, can be calculated by

$$
\begin{aligned}
MD_k(0,0) \quad = \quad & 5 \times 2^{k-1} - 2 - 2a_{k-1} + MD_{k-1}(0,0) + \\
& 3MD_{k-1}(E_{k-1}), \qquad k \ge 2,
\end{aligned} \tag{8}
$$

and $MD_1(0,0) = 3$. The following is the exact expression for Eq. (8),

$$MD_k(0,0) = \frac{6}{5} \times 2^{2k} + \frac{4}{3} \times 2^k - 2k - \frac{1}{30} \times (-1)^k - \frac{5}{2}, \tag{9}$$

where $k \ge 1$. For examples, $MD_2(0,0) = 18$, $MD_3(0,0) = 79$, and $MD_4(0,0) = 318$. These $MD_k(0,0)$s match the results of the previous examples in Section 2. The detailed derivation of Eq. (9) can be found in [1].

**Definition 4**: *In an $n \times n$ mesh ($n = 2^k$), the optimal $TCD$ is defined as $min\{MD_k(x, y)\}$, where $1 \leq x, y \leq n - 1$.*

From Theorems 1 and 2, we immediately have the following corollary.

**Corollary 1**: *If we start a broadcast from an eye of a mesh and follow Algorithm 1, the $TCD$ obtained is an optimal $TCD$, i.e., $MD_k(E_k) = min\{MD_k(x, y)\}$.*

The optimal $TCD$ broadcast for a $2^k \times 2^k$ mesh is just a special case of Algorithm 1, in which the broadcast originates from an eye of a mesh. In this case, the source node sends the message to eye $E_k(3)$ in the second step, since the source node is in region $(i)$ (see Fig.6).

The recursive formula for $MD_k(E_k)$ is the same as Eq. (3). The exact expression can be derived as follows:

$$MD_k(E_k) = \frac{1}{5}[3 \times 2^{2k+1} - (-1)^k] - 2^k, \qquad k \geq 1\ . \quad (10)$$

For examples, $MD_1(E_1) = 3$, $MD_2(E_2) = 15$, $MD_3(E_3) = 69$ and $MD_4(E_4) = 291$. These $MD_k(E_k)$s match the results of the previous examples in Section 2. The detailed derivation of Eq. (10) can be found in [1].

## 4. Minimizing $TCD$ of a Time-Step Optimal Broadcast in a $d$-D Mesh

A $\overbrace{2^k \times 2^k \times \ldots \times 2^k}^{d}$ mesh is also called a $d$-D $2^k$ mesh, or simply a $d$-D $2^k$ mesh. The definition of an eye in such a mesh is defined as follows:

**Definition 5**: *There are $2^d$* eyes *in a $d$-D $2^k$ mesh, labeled as $E_k^d(i)$, $0 \leq i \leq 2^d - 1$. These eyes are recursively defined as follows: All $2^d$ nodes of a $d$-D $2^1$ mesh are eyes, $E_1^d(i)$. A $d$-D $2^k$ mesh is partitioned into $2^d$ $d$-D $2^{k-1}$ submeshes, each of which has $2^d$ eyes, $E_{k-1}^d(i)$. Eyes $E_k^d(i)$ are selected from $2^{2d}$ $E_{k-1}^d(i)$s. Specifically, eyes $E_k^d(i)$ are the $2^d$ $E_{k-1}^d(i)$s that are the closest to the center of the $d$-D $2^k$ mesh.*

For example, a $d$-D $2^2$ mesh consists of $2^d$ $d$-D $2^1$ submeshes, each of which has $2^d$ $E_1^d(i)$s, $0 \leq i \leq 2^d - 1$. Among all these $2^{2d}$ $E_1^d(i)$s, the inner $2^d$ ones, which are the closest to the center of the $d$-D $2^2$ mesh, are the eyes of the $d$-D $2^2$ mesh, $E_2^d(i)$. Sometimes, we use $E_k^d$ to represent $E_k^d(i)$ to simplify our notation. Here we restrict our attention only to the cases where the source node is an eye of a $d$-D mesh.

**Definition 6**: *Denote $MD_k^d(E_k)$ as the minimum $TCD$ for a $d$-D $2^k$ mesh to complete a broadcast from an eye.*

Let's look at some examples of 3-D meshes. A $2 \times 2 \times 2$ mesh is 3-D unit mesh. Fig.7 (b) shows the process of a broadcast in a $2 \times 2 \times 2$ mesh. The optimal $TCD$ is

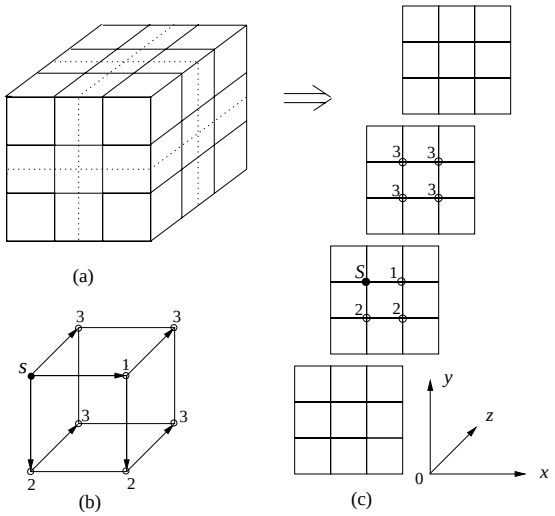


**Figure 7. (a) and (c) Broadcast in a $4 \times 4 \times 4$ mesh. (b) Broadcast in a $2 \times 2 \times 2$ mesh.**

$MD_1^3(E_1) = FD + SD + TD = 2^0 + 2^1 + 2^2 = 7$, where $TD$ is denoted as the total distance of the third step . It takes three steps to complete the broadcast with each step responsible for one dimension and the number of the nodes to be delivered doubles in each step. Using the same way for a $d$-D unit mesh, we immediately deduce that it takes $d$ steps to complete a broadcast in a $d$-D unit mesh. The optimal $TCD$ is

$$MD_1^d(E_1) = \sum_{i=0}^{d-1} 2^i = 2^d - 1. \quad (11)$$

Fig.7 (a) and (c) show the process of a broadcast in a $4 \times 4 \times 4$ mesh from an eye. Fig.7 (a) shows that the $4 \times 4 \times 4$ mesh consists of eight $2 \times 2 \times 2$ submeshes (unit meshes) and Fig.7 (c) shows a partition of the $4 \times 4 \times 4$ mesh into four $4 \times 4$ submeshes along dimension $z$. During the first three steps, the message is delivered to eight eyes including the source. All these eyes are located in different $2 \times 2 \times 2$ submeshes. In the remaining three steps, eight submeshes complete delivering the message within themselves following the same process as shown in Fig.7 (b). The optimal $TCD$ is $MD_2^3(E_2) = (2^0 + 2^1 + 2^2)a_2 + 2^3 MD_1^3(E_1) = 63$. In the same way, it is easy to obtain the optimal $TCD$ for a $8 \times 8 \times 8$ mesh, $MD_3^3(E_3) = (2^0 + 2^1 + 2^2)a_3 + 2^3 MD_2^3(E_2) = 525$.

We can extend our optimal $TCD$ algorithm for a 2-D mesh to a $d$-D mesh. In a $d$-D $2^k$ mesh, it needs totally $\log n^d = dk$ steps to complete a broadcast. We divide these $dk$ steps into $k$ phases, each of which consists of $d$ steps. In the first phase, the first $d$ steps are for the broadcast among

all the $2^d$ eyes of $d$-D $2^k$ meshes. In the second phase, the next $d$ steps are for the broadcast among all the $2^{2d}$ eyes of $2^d$ $d$-D $2^{k-1}$ submeshes. In the last phase (the $k$th phase), the last $d$ steps are for the broadcast among all the $2^{kd}$ eyes of $2^{(k-1)d}$ $d$-D $2^1$ submeshes (unit meshes). If we start a broadcast from an eye of a $d$-D mesh and follow the above extended algorithm, we will obtain an optimal $TCD$ (see [1] for details). Also, by extending our minimum $TCD$ algorithm for a 2-D mesh to a $d$-D mesh, we can obtain the minimum $TCD$ for a given source node.

The recursive formula for optimal $TCD$ of a $d$-D $2^k$ mesh is

$$\begin{aligned} MD_k^d(E_k) &= \sum_{i=0}^{d-1} 2^i a_k + 2^d MD_{k-1}^d(E_{k-1}) \\ &= (2^d - 1)a_k + 2^d MD_{k-1}^d(E_{k-1}), \quad (12) \end{aligned}$$

where $k \geq 2$ and $MD_1^d(E_1)$ is given by Eq. (11). The exact expression is

$$\begin{aligned} MD_k^d(E_k) &= \tfrac{2^d-1}{3(2^d+1)(2^{d-1}-1)}[3 \times 2^{d(k+1)-1} - \\ &\quad (-1)^k(2^{d-1}-1) - 2^k(2^d+1)], \quad (13) \end{aligned}$$

where $k, d \geq 1$. The detailed derivations can be found in [1]. For example, when $d = 2$, Eq. (13) leads to the minimum $TCD$ expression for a 2-D mesh which is exactly the same as Eq. (10). When $d = 3$, Eq. (13) leads to the minimum $TCD$ expression for a 3-D mesh.

$$MD_k^3(E_k) = \frac{7}{27}[2^{3k+2} - (-1)^k - 3 \times 2^k], \quad k \geq 1\ . \quad (14)$$

## 5. Conclusion

In this paper we have identified a set of special nodes called eyes in a given mesh. Both the minimum $TCD$ broadcast algorithm from a given source and the optimal $TCD$ broadcast algorithm (which is the minimum one among $TCD$s for all the possible source nodes) are based on the idea of eyes. If we start a broadcast from a given source node and follow the minimum $TCD$ algorithm, a minimum $TCD$ from the source is obtained, which is the minimum one among all the possible $TCD$s from this given source node. If we start a broadcast from an eye of a mesh and follow the minimum $TCD$ algorithm, an optimal $TCD$ is obtained.

Our results can be easily extended to a torus, which is a special mesh in which the nodes at the periphery are connected by wraparound connections. This means that each node in the torus is equivalent to each other, i.e., each node in a torus is an eye in the corresponding mesh. Therefore, no matter where a broadcast in the torus is initiated, we can always use the proposed minimum $TCD$ algorithm and obtain an optimal $TCD$.

## References


[1] S. Cang and J. Wu, "Minimizing total communication distance of a broadcast on mesh and torus networks", *TR-CSE-97-57*, Florida Atlantic University, September 1997.

[2] W. J. Dally, "The J-machine: system support for actors", *Actors: Knowledge-based Concurrent Computing*, Hewitt and Agha, Eds., MIT Press, 1989.

[3] J. Duato, S. Yalmanchili, and L. Ni, *Interconnection Networks: An Engineering Approach*, IEEE Computer Society, 1997.

[4] X. Lin and L. M. Ni, "Multicast communication in multicomputers networks", *IEEE Transactions on Parallel and Distributed Systems*, 4, (10), October 1993, 1105-1117.

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

[6] I. Wojciechowska, "Line broadcast in grid graphs", *The 28th Southeastern International Conference on Combinatorics Graph Theory Computing*, March 1997, (abstract only).


## Appendix

(Proof of Theorem 2)

We prove this theorem using the induction on k. Assume that the source node is represented by $S$. Theorem 2 can be proved by showing

$$D'_l(S) - D_l(S) \geq 0 \quad (15)$$

to be true for any integer $l$, where $D_l(S)$ is the $TCD$ obtained from Algorithm 1 and $D'_l(S)$ is the $TCD$ obtained from an arbitrarily selected broadcast algorithm.

For $l = 1$, it is a $2 \times 2$ mesh. Algorithm 1 is the only possible approach and hence $D'_1(S) = D_1(S)$. For $l = 2$, it is a $4 \times 4$ mesh. When $S = (0,0)$, $D_2(0,0) = 18$, all the other possible $D_2(0,0)$s are greater than or equal to 18. When $S = (1,0)$ or $(0,1)$, $D_2(1,0) = D_2(0,1) = 17$, all the other possible $D'_2(1,0)$s or $D'_2(0,1)$s are greater than or equal to 17. When $S = (1,1)$, $D_2(1,1) = 15$, all the other possible $D'_2(1,1)$s are greater than 15. Therefore, $D_2(S) = MD_2(S)$, i.e., Eq. (15) is true. For $l = k-1$, assume that up to this level Theorem 2 is true, i.e., $D'_l(S) - D_l(S) \geq 0$, $1 \leq l \leq k-1$, and $D_{k-1}(s) = MD_{k-1}(S)$.

For $l = k$, in order to prove $D'_k(S) - D_k(S) \geq 0$, we need to determine $D'_k(S)$, which is the $TCD$ obtained

from an arbitrarily selected broadcast algorithm. We denote the three destination nodes of the first two steps as $N_2$, $N_3$, and $N_4$, respectively. Their coordinates $(u_{k-1}(i), v_{k-1}(i))$, $i = 2, 3, 4$, are indicated in Fig.5. Note that the coordinate system of each $2^{k-1} \times 2^{k-1}$ submesh is set up according to the convention in Fig.4. Following Algorithm 1, the first step is from $S(u_k, v_k)$ to $N_2(u_{k-1}(2), v_{k-1}(2))$,

$$FD = a_k + u_k - v_{k-1}(2) + |v_k + u_{k-1}(2)|.$$

The second step is from $S$ to $N_3(u_{k-1}(3), v_{k-1}(3))$ and from $N_2(u_{k-1}(2), v_{k-1}(2))$ to $N_4(u_{k-1}(4), v_{k-1}(4))$,

$$\begin{aligned} SD = {} & a_k + v_k - u_{k-1}(3) + |u_k + v_{k-1}(3)| + a_k \\ & -u_{k-1}(2) - v_{k-1}(4) + |v_{k-1}(2) - u_{k-1}(4)|. \end{aligned}$$

The remaining step, $RD$, is calculated by

$$RD = \sum_{i=1}^{4} MD_{k-1}(N_i),$$

where $MD_{k-1}(S = N_1)$, $MD_{k-1}(N_2)$, $MD_{k-1}(N_3)$, and $MD_{k-1}(N_4)$ can be obtained from Eq. (7), i.e.,

$$MD_{k-1}(N_i) = MD_{k-1}(E_{k-1}) + \sum_{j=2}^{k-1} f_j(N_i),$$

where $i = 1, 2, 3, 4$. Therefore,

$$\begin{aligned} D'_k(S) = {} & u_k - u_{k-1}(2) - u_{k-1}(3) + v_k - v_{k-1}(2) - v_{k-1}(4) \\ & +|v_k + u_{k-1}(2)| + |u_k + v_{k-1}(3)| + |v_{k-1}(2) - u_{k-1}(4)| \\ & +3a_k + 4MD_{k-1}(E_{k-1}) + \sum_{i=1}^{4}\sum_{j=2}^{k-1} f_j(N_i) \end{aligned} \quad (16)$$

From Eq. (3) and (7), we have

$$D_k(S) = 3a_k + 4MD_{k-1}(E_{k-1}) + MD_k(E_k) + \sum_{j=2}^{k} f_j(S)\ .$$

Subtracting Eq. (16) by above equation, we have

$$D'_k(S) - D_k(S) = \Delta_1 + \Delta_2 + \Delta_3 + \Delta_4 + \Delta_5, \quad (17)$$

where

$$\begin{aligned} \Delta_1 = {} & u_k + v_k - f_k(S) + |u_k + v_{k-1}(3)| + |v_k + u_{k-1}(2)| \\ \Delta_2 = {} & -u_{k-1}(2) - v_{k-1}(2) + \textstyle\sum_{j=2}^{k-1} f_j(N_2) \\ \Delta_3 = {} & -u_{k-1}(3) + \textstyle\sum_{j=2}^{k-1} f_j(N_3) \\ \Delta_4 = {} & -v_{k-1}(4) + \textstyle\sum_{j=2}^{k-1} f_j(N_4) \\ \Delta_5 = {} & |v_{k-1}(2) - u_{k-1}(4)|. \end{aligned}$$

It is obvious that $\Delta_5 \geq 0$. In $\Delta_2$, $\sum_{j=2}^{k-1} f_i(N_2)$ is always positive (see Eq. (6)). When $u_{k-1}(2)$, $v_{k-1}(2) \leq 0$, $\Delta_2 \geq 0$ automatically. When $u_k, v_k \geq 0$, $f_{k-1}(u_{k-1}(2), v_{k-1}(2))$ is either $2v_{k-1}(2) + u_{k-1}(2) + |u_{k-1}(2) - a_{k-2}|$ or $2v_{k-1}(2) + 2u_{k-1}(2)$ (see Eq. (5)), which makes $\Delta_2 \geq 0$. Therefore, $\Delta_2 \geq 0$ is always positive. In the same way, we can show that both $\Delta_3$ and $\Delta_4$ are always positive. Specifically, $\Delta_3 > -u_{k-1}(3) + f_{k-1}(N_3) > 0$, and $\Delta_4 > -v_{k-1}(4) + f_{k-1}(N_4) > 0$.

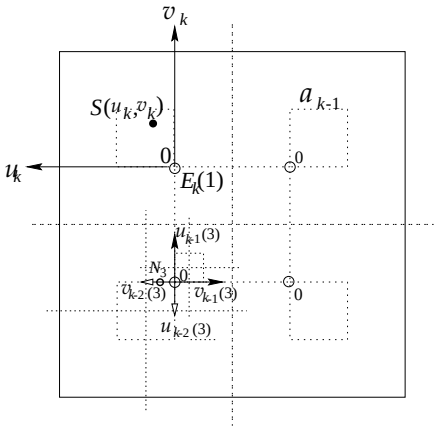


**Figure 8. Calculation of $f_{k-2}(N_3)$.**

$\Delta_1$ in Eq. (17) is rather complex because $f_k(S)$ has six different expressions corresponding to six different locations of $S$. Further more, it depends on the locations of $N_2$ and $N_3$. Here we just consider one case (region $(iv)$ in Fig.6). The remaining cases can be treated similarly [1]. For the case of $0 < u_k \leq a_{k-1}/2$, and $0 < v_k \leq 2^{k-1} - a_k/2$, $f_k(S) = 2u_k + 2v_k$. Thus,

$$\Delta_1 = -u_k - v_k + |u_k + v_{k-1}(3)| + |v_k + u_{k-1}(2)|.$$

When both $u_{k-1}(2) > 0$ and $v_{k-1}(3) > 0$, $\Delta_1$ is obviously greater than zero. But when both $u_{k-1}(2) < 0$ and $v_{k-1}(3) < 0$, $\Delta_1$ is less than zero. We have to find other terms to nullify this negative term. Actually, there are many positive terms in both $\Delta_2$ and $\Delta_3$. For example, $f_{k-2}(N_3)$ hasn't been used when we showed that $\Delta_3 > 0$. Therefore, we can use $f_{k-2}(N_3)$ to nullify the negative value of $\Delta_1$. Specifically, when $v_{k-1}(3) < 0$, as shown in Fig.8, $v_{k-2}(3) = -v_{k-1}(3) > 0$. Thus $f_{k-2}(N_3) \geq 2|v_{k-1}(3)|$. We also use a term from $\Delta_2$ to nullify the negative value of $\Delta_1$. It can be seen that $\Delta_2 > |u_{k-1}(2)|$ when $u_{k-1}(2) < 0$. Therefore, Eq. (17) can be written as

$$\begin{aligned} D'_k(S) - D_k(S) > {} & \Delta_1 + \Delta_2 + \Delta_3 \\ > {} & \Delta_1 + |u_{k-1}(2)| + 2|v_{k-1}(3)| \\ = {} & (-|u_k| + |u_k + v_{k-1}(3)| + 2|v_{k-1}(3)|) + \\ & (-|v_k| + |v_k + u_{k-1}(2)| + |u_{k-1}(2)|). \end{aligned}$$

It is not difficult to see that $-|u_k| + |u_k + v_{k-1}(3)| + 2|v_{k-1}(3)| > 0$ and $-|v_k| + |v_k + u_{k-1}(2)| + |u_{k-1}(2)| > 0$. Therefore, $D'_k(S) - D_k(S) > 0$.

We can show that $D'_k(S) - D_k(S) > 0$ is also true for all the other cases (corresponding to other five regions in Fig.6). Therefore, Eq. (15) is true for $l = k$. In summary, Eq. (15) is true for all $l$, i.e., Theorem 2 is valid. □