Journal of Parallel and Distributed Computing **60**, 966–997 (2000)
doi:10.1006/jpdc.2000.1642, available online at http://www.idealibrary.com on IDEAL®

Time-Step Optimal Broadcasting in 3-D Meshes with Minimum Total Communication Distance[1]

Songluan Cang and Jie Wu

Department of Computer Science and Engineering, *Florida Atlantic University*, *Boca Raton*, *Florida 33431*
E-mail: scang@cse.fau.edu, jie@cse.fau.edu

Received October 1, 1998; revised March 2, 2000; accepted March 7, 2000

In this paper we propose a new minimum *total communication distance* (*TCD*) algorithm and an optimal *TCD* algorithm for broadcast in a 3-dimensional mesh (3-D 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 are based on a divide-and-conquer approach where a 3-D mesh is partitioned into eight submeshes of equal size. The source node sends the broadcast message to a special node called an *eye* in each submesh. The above procedure is then recursively applied in each submesh. The proposed approach can be generalized to a *d*-dimensional mesh or torus. In addition, the proposed approach can potentially be used to solve optimization problems in other collective communication operations. © 2000 Academic Press

Key Words: broadcast; communication distance; divide-and-conquer; meshes; optimization problems; wormhole routing.

1. INTRODUCTION

In a multicomputer system, a collection of processors (also called nodes) work together to solve large applications. *Mesh-connected topology* is one of the most thoroughly investigated network topologies for multicomputer systems. It is important due to its simple structure and its good performance in practice and is becoming popular for reliable and high-speed communication switching. Mesh-connected topologies, also called *k*-ary *d*-dimensional meshes, have a *d*-dimensional grid structure with *k* nodes in each dimension such that every node is connected to two other nodes in each dimension by a direct communication. Mesh-connected topologies include *n*-dimensional meshes, tori, and hypercubes. These topologies have desirable properties of regularity, balanced behavior, and a large number of alternative paths. Machines that use 2-dimensional (2-D) meshes includes the MIT J-machine [2], the Symult 2010 [14], and the Intel Touchstone Delta [7]. The Cray T3E [5] system uses a 3-D (3-dimensional) torus.

[1] This work was supported in part by NSF grant 9900646.

0743-7315/00 $35.00

In order to minimize communication latency it is important to design an efficient implementation of collective communication operations [11, 12] which include multicast and broadcast. *Multicast* is an important system-level communication service [3, 8] in which the same message is delivered from a source to an arbitrary number of destination nodes. *Broadcast* [4] 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 [6], clock synchronization [13], and compute-aggregate-broadcast type of algorithms [10].

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* is important in designing an efficient broadcast. A *minimum TCD* algorithm for broadcast from a given source node is the one that generates a 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 3-D mesh, say an $n \times n \times n$ mesh with $n = 2^k$, where k is a nonnegative integer, we only consider broadcast algorithms that can complete a broadcast in $\log n^3 = 3k$ time steps in a wormhole-routed system. The *wormhole switching* technique is becoming the trend in building multicomputer systems due to its inherent advantages such as low-latency communication and reduced communication hardware overhead. Under the wormhole switching [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 the system under consideration uses the one-port model, i.e., at each time step a node may do one of the following: send a message to one node, receive a message from one node, or be idle. Note that without the minimum *TCD* requirement, time-step optimal 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 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.

In general, *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 (link). Another contention is called *depth contention* which 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. Note that in a time-step optimal broadcast algorithm, the source node sends $3k$ copies at different steps. Since there are only six adjacent links, depth contention is unavoidable for $k > 2$. Therefore, the issue of depth contention–freedom will not be further discussed. Notice that when

each node in a 3-D mesh is synchronized and the broadcast message is relatively short, step contention–freedom implies depth contention–freedom.

The divide-and-conquer approach is applied to achieve a minimum *TCD* broadcasting where a given 3-D mesh is partitioned into eight submeshes of equal size. We identify a set of eight special nodes called *eyes* in a given 3-D mesh. The source node sends the broadcast message to an *eye* in each submesh. The optimization problem is then solved recursively at each submesh (each submesh has its own eight eyes). Specifically, we propose a minimum *TCD* broadcast algorithm for a given source node and an optimal *TCD* broadcast algorithm. 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 optimal *TCD* algorithm, an optimal *TCD* is obtained, which is the minimum one among *TCD*s for all the possible source nodes.

In summary, our approach is surprisingly simple. For any given source node, the time-step optimal broadcast that achieves a minimum *TCD* always forwards the broadcast message to several fixed locations (eyes) in a predefined order. This process is independent of the location of the source. When a given source node is an eye itself, the corresponding broadcast generates an optimal *TCD*.

There are several related works, but they focus on either broadcasting under the all-port model which supports simultaneous send and receive to and from all neighbors [17, 18] or different communication patterns such as complete exchange [16] or all-to-all personalized exchange [15]. Most of these works focus on minimizing time-step, since it is no longer a trivial problem, without considering minimizing total communication distance. The only previous work that considers minimizing total communication distance is the one by Wojciechowska [19]; however, the source of broadcasting is restricted to a corner of a 2-D mesh.

The remainder of the paper is organized as follows. Section 2 introduces necessary notations and preliminaries, where the concept of eyes in a 2-D mesh is reviewed. Both the minimum *TCD* broadcast algorithm and the optimal *TCD* broadcast algorithm in a 2-D mesh are also reviewed. Section 3 provides our major results on *TCD*s for 3-D meshes, where the eyes of a 3-D mesh are defined and a minimum *TCD* broadcast algorithm and an optimal *TCD* broadcast algorithm in a 3-D mesh are proposed. A closed form expression for the optimal *TCD* in a 3-D mesh is provided. In Section 4, we conclude this paper and discuss possible future work. The proof of a major result (Theorem 3) is included in the Appendix of the paper.

2. PRELIMINARIES

In this section, we briefly review the major result in [1], which is about minimizing the *TCD* of a time-step optimal broadcast in 2-D meshes. 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), where $x, y = 0, 1, 2, ..., n-1$. A node N at (x, y) is denoted by $N(x, y)$. The origin of the coordinate system is the upper-left corner of the mesh, as shown in Fig. 1.

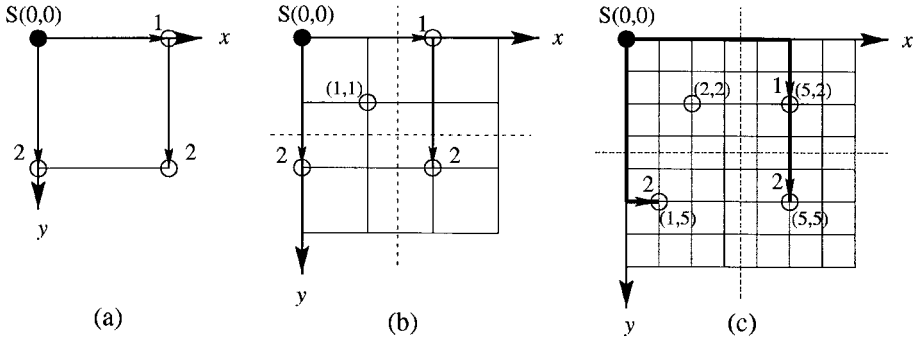

FIG. 1. (a) Broadcast in a 2×2 mesh. (b) Broadcast in a 4×4 mesh. (c) Broadcast in an 8×8 mesh.

Denote $D_k^2(x, y)$ or $D_k^2(S)$ as the *TCD* of a broadcast algorithm originated from a source node $S(x, y)$ in a $2^k \times 2^k$ mesh and $MD_k^2(x, y)$ or $MD_k^2(S)$ as the minimum *TCD* originated from a source node $S(x, y)$ among all the possible broadcast algorithms. Obviously, $MD_k^2(x, y) = \min\{D_k^2(x, y)\}$. Denote OD_k^2 as the optimal *TCD* in a $2^k \times 2^k$ mesh. Clearly, $OD_k^2 = \min\{MD_k^2(x, y)\}$. Denote *FD* as the communication distance in the first step of a broadcast, *SD* as the total communication distance in the second step, and *RD* as the total communication distance in the remaining steps. Obviously, for a given source node $S(x, y)$ in a mesh, $D_k^2(x, y) = FD + SD + RD$ is the basic formula to calculate the *TCD* for a particular broadcast algorithm.

For example, Figs. 1a, 1b, and 1c show the processes of broadcast starting from node $S(0, 0)$ in a 2×2 mesh, a 4×4 mesh, and an 8×8 mesh, respectively. Arrows 1 and 2 represent the first and second steps of broadcast, respectively. In Fig. 1a, the *TCD* is $D_1^2(0, 0) = FD + SD = 1 + (1 + 1) = 3$. Obviously, the same result will be obtained wherever the broadcast starts in this mesh. This means that $OD_1^2 = MD_1^2(x, y) = D_1^2(x, y) = 3$, where $x, y = 0$ or 1. In Fig. 1b, $FD = 2$, $SD = 2 + 2 = 4$, and $RD = 3 + 3 + 3 + 3 = 12$. Therefore, the *TCD* is $D_2^2(0, 0) = FD + SD + RD = 2 + 4 + 12 = 18$. $D_2^2(0, 0) = 18$ turns out to be the minimum *TCD* by comparing it with results of all the other arrangements; that is, $MD_2^2(0, 0) = 18$. The minimum *TCD*s for other nodes in the 4×4 mesh can be easily obtained in the same way, $MD_2^2(1, 0) = 1 + (2 + 2) + (3 + 3 + 3 + 3) = 17$, $MD_2^2(0, 1) = 2 + (1 + 1) + (3 + 3 + 3 + 3) = 16$, and $MD_2^2(1, 1) = 1 + (1 + 1) + (3 + 3 + 3 + 3) = 15$. Obviously, when the source node is at $(1, 1)$, $OD_2^2 = MD_2^2(1, 1) = 15$. In Fig. 1c, $FD = 7$, $SD = 6 + 3 = 9$, and $RD = M_2^2(0, 0) + 3 \times MD_2^2(1, 1) = 18 + 3 \times 15 = 63$. Therefore, the *TCD* is $D_3^2(0, 0) = FD + SD + RD = 7 + 9 + 63 = 79$. This is actually the best solution for a corner node, i.e., $MD_3^2(0, 0) = 79$. It is not difficult to determine that the optimal *TCD* for an 8×8 mesh is 69 when the source node is at $(2, 2)$; i.e., $OD_3^2 = MD_3^2(2, 2) = 69$.

2.1. Eyes of a 2-D mesh

DEFINITION 1 [1]. There are four eyes in a $2^k \times 2^k$ mesh, labeled as $E_k^2(i)$, $i = 1, 2, 3, 4$. These eyes are recursively defined as follows: All four nodes in a 2×2 mesh are eyes, $E_1^2(i)$, $i = 1, 2, 3, 4$, as shown in Fig. 2a. A $2^k \times 2^k$ mesh is partitioned

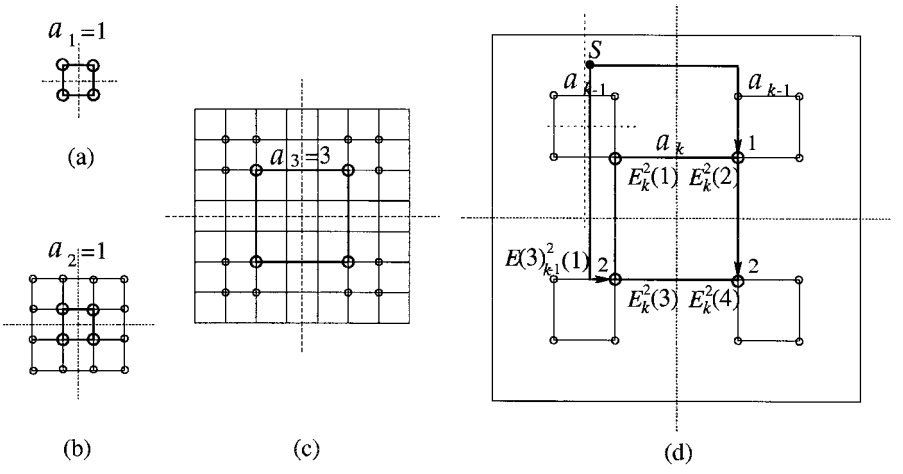

FIG. 2. The recursive definition of eyes of (a) a 2×2 mesh, (b) a 4×4 mesh, (c) an 8×8 mesh, and (d) a $2^k \times 2^k$ mesh.

into four $2^{k-1} \times 2^{k-1}$ submeshes, each of which has four eyes, $E(i)^2_{k-1}(j)$, $i, j = 1, 2, 3, 4$. Eyes, $E^2_k(i)$, are selected from sixteen $E(i)^2_{k-1}(j)$s. Specifically, $E^2_k(i)$, $i = 1, 2, 3, 4$, belong to the upper-left, upper-right, lower-left, and lower-right submesh, respectively, and they are the four $E(i)^2_{k-1}(j)$s that are the closest to the center of the $2^k \times 2^k$ mesh among the sixteen $E(i)^2_{k-1}(j)$s, as shown in Fig. 2d.

For example, the inner four nodes of a 4×4 mesh as shown in Fig. 2b are eyes, $E^2_2(i)$, $i = 1, 2, 3, 4$. Fig. 2c shows four eyes of an 8×8 mesh, $E^2_3(i)$, $i = 1, 2, 3, 4$. We denote $i = 1, 2, 3, 4$ as the indices of the upper-left, upper-right, lower-left, and lower-right submeshes and eyes, respectively. We also use E^2_k to represent $E^2_k(i)$ to simplify our notation when there is no need to distinguish these four eyes.

Define the square, formed by four eyes $E^2_k(i)$ of a $2^k \times 2^k$ mesh as its four corners, as the *eye-square* of the $2^k \times 2^k$ mesh. Denote a_k as the length of the side of this eye-square. Denote $(x_k(i), y_k(i))$ as the coordinates of $E^2_k(i)$, $i = 1, 2, 3, 4$, respectively. a_k is calculated by

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

For example, $a_1 = 1$, $a_2 = 1$, $a_3 = 3$, $a_4 = 5$, and $a_5 = 11$, etc. Using Eq. (1), we can easily determine locations of four eyes of a given $2^k \times 2^k$ mesh. Specifically,

$$x_k(1, 3) = \tfrac{1}{6}[2^{k+1} + (-1)^k] - \tfrac{1}{2}, \qquad y_k(1, 2) = x_k(1),$$
$$x_k(2, 4) = \tfrac{1}{6}[2^{k+2} - (-1)^k] - \tfrac{1}{2}, \qquad y_k(3, 4) = x_k(2).$$

For example, the coordinates of four eyes of a 2×2 mesh are (0, 0), (1, 0), (0, 1), and (1, 1). The coordinates of four eyes of a 4×4 mesh are (1, 1), (2, 1), (1, 2), and (2, 2). The coordinates of four eyes of an 8×8 mesh are (2, 2), (5, 2), (2, 5), and (5, 5).

2.2. Minimum TCD Broadcast Algorithm in a 2-D Mesh

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

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 S is in the upper-left submesh, as shown in Fig. 2d.

2. The source node sends the message to the upper-right eye $E_k^2(2)$ in the first step.

3. In the second step, $E_k^2(2)$ sends the message to the lower-right eye $E_k^2(4)$, as shown in Fig. 2d, and the source node sends the message to either the lower-left eye $E_k^2(3)$ or $E(3)_{k-1}^2(1)$ depending on which one is closer to the source node. That is, if the source node is in the right-half region of the submesh, it sends the message to $E_k^2(3)$, as shown in Fig. 2d; if it is in the left-half region of the submesh, it sends the message to $E(3)_{k-1}^2(1)$.

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 submesh's level by level until reaching the unit meshes, 2×2 meshes, and all these unit meshes complete the broadcast within themselves in two steps, as shown in Fig. 1a.

Algorithm 1 [1] is a minimum *TCD* broadcast algorithm in a 2-D mesh and the following two major results are also shown in [1]:

1. If the source node is an eye of the mesh, the TCD obtained by applying Algorithm 1 is the minimum among results obtained by applying Algorithm 1 to all the possible source nodes.

2. The TCD obtained by applying Algorithm 1 is the minimum TCD for a given source node.

In an $n \times n$ mesh ($n = 2^k$), the optimal *TCD* is defined as $OD_k^2 = \min\{MD_k^2(x,y)\}$, where $1 \leqslant x, y \leqslant n-1$. The optimal *TCD* broadcast algorithm 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. Based on the above two major results, the *TCD* obtained by applying Algorithm 1 to an eye of the mesh is the optimal *TCD*; i.e., $OD_k^2 = MD_k^2(E_k) = \min\{MD_k^2(x, y)\}$. In the remainder of the paper, we use $MD_k^2(E_k^2)$ to represent the optimal *TCD*. The recursive formula for $MD_k^2(E_k^2)$ is

$$MD_k^2(E_k^2) = 3a_k + 4MD_{k-1}^2(E_{k-1}^2), \tag{2}$$

where $MD_1^2(E_1^2) = 3$. The optimal *TCD* of a $2^k \times 2^k$ mesh can be calculated by

$$MD_k^2(E_k^2) = \tfrac{1}{5}[3 \times 2^{2k+1} - (-1)^k] - 2^k, \qquad k \geqslant 1. \tag{3}$$

For example, $MD_1^2(E_1^2) = 3$, $MD_2^2(E_2^2) = 15$, $MD_3^2(E_3^2) = 69$, $MD_4^2(E_4^2) = 291$, and $MD_5^2(E_5^2) = 1197$.

3. MINIMIZING *TCD* OF A TIME-STEP OPTIMAL BROADCAST IN A 3-D MESH

In this section, we discuss time-step optimal broadcasting with minimum *TCD* in a 3-D mesh by extending the results for a 2-D mesh to a 3-D mesh. A $2^k \times 2^k \times 2^k$ mesh is also called a 3-dimensional 2^k mesh, or simply a 3-D 2^k mesh. The eyes of a 3-D 2^k mesh are defined in the following section.

3.1. Eyes of a 3-D mesh

For a given 3-D 2^k mesh, we assume that the distance between any two adjacent nodes is one. The location of a node in a 3-D mesh is denoted by a set of coordinates (x, y, z), where $x, y, z = 0, 1, 2, ..., 2^k - 1$. Without loss of generality, the origin of the $x-y-z$ coordinate system is assumed to be a corner node of the mesh, as shown in Fig. 3. A node N at (x, y, z) is denoted by $N(x, y, z)$.

DEFINITION 2. There are $2^3 = 8$ eyes in a 3-D 2^k mesh, labeled as $E_k^3(i)$, where $i = 1, 2, ..., 8$ are called the indices of eight eyes. These eyes are recursively defined as follows: All eight nodes of a 3-D 2^1 mesh (i.e., $2 \times 2 \times 2$ mesh) are eyes, $E_1^3(i)$, as shown in Fig. 3a. A 3-D 2^k mesh is partitioned into eight 3-D 2^{k-1} submeshes, each is labeled as the ith submesh where the eye $E_k^3(i)$ locates, and $i = 1, 2, ..., 8$ are also called the indices of eight submeshes. Each submesh has eight eyes, $E(i)_{k-1}^3(j)$, $i, j = 1, 2, ..., 8$. Eyes $E_k^3(i)$ are selected from a total of $8 \times 8 = 64$ $E(j)_{k-1}^3(i)$s. Specifically, eyes $E_k^3(i)$ are the eight $E(j)_{k-1}^3(i)$s that are the closest to the center of the 3-D 2^k mesh, as shown in Fig. 3c.

For example, a 3-D 2^2 mesh (i.e., $4 \times 4 \times 4$ mesh) consists of $2^3 = 8$ 3-D 2^1 submeshes (i.e., $2 \times 2 \times 2$ submeshes), each of which has eight eyes, $E(j)_1^3(i)$s, $i, j = 1, 2, ..., 8$. Among these $8 \times 8 = 64$ $E(j)_1^3(i)$s, the inner most eight eyes, which are the

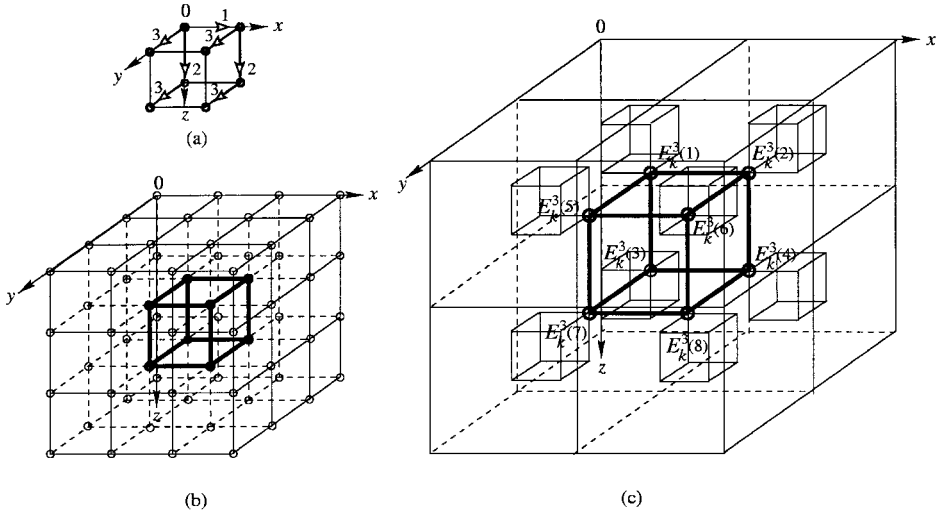

FIG. 3. The recursive definition of eyes of (a) a $2 \times 2 \times 2$ mesh, (b) a $4 \times 4 \times 4$ mesh, and (c) a $2^k \times 2^k \times 2^k$ mesh.

closest to the center of the 3-D 2^2 mesh, are the eyes of the 3-D 2^2 mesh, $E_2^3(i)$, as shown in Fig. 3b. We also use E_k^3 to represent $E_k^3(i)$s to simplify our notation when there is no need to distinguish these eight eyes.

DEFINITION 3. Define the cube, formed by eight eyes $E_k^3(i)$ of a 3-D 2^k mesh as its eight corners, as the *eye-cube* in the 3-D 2^k mesh. Denote a_k^3 as the length of the side of this eye-cube. Denote $(x_k(i), y_k(i), z_k(i))$, $i = 1, 2, ..., 8$, as the coordinates of eight $E_k^3(i)$, respectively.

Obviously, the length of the side of the eye-cube in the 3-D 2^k mesh is equal to the length of the side of the eye-square of the $2^k \times 2^k$ mesh, which is calculated by Eq. (1). Therefore, the length of the side of the eye-cube can be calculated by

$$a_k^3 = \tfrac{1}{3}[2^k - (-1)^k], \qquad k \geqslant 1. \tag{4}$$

Using Eq. (4), we can easily determine the locations of eight eyes of a given 3-D 2^k mesh. Specifically,

$$x_k(1,3,5,7) = \tfrac{1}{6}[2^{k+1} + (-1)^k] - \tfrac{1}{2}, \quad y_k(1,2,3,4) = x_k(1), \quad z_k(1,2,5,6) = x_k(1),$$
$$x_k(2,4,6,8) = \tfrac{1}{6}[2^{k+2} - (-1)^k] - \tfrac{1}{2}, \quad y_k(5,6,7,8) = x_k(2), \quad z_k(3,4,7,8) = x_k(2).$$

For example, the coordinates of eight eyes of a $2 \times 2 \times 2$ mesh are (0, 0, 0), (1, 0, 0), (0, 0, 1), (1, 0, 1), (0, 1, 0), (1, 1, 0), (0, 1, 1), and (1, 1, 1) (see Fig. 3a). The coordinates of eight eyes of a $4 \times 4 \times 4$ mesh are (1, 1, 1), (2, 1, 1), (1, 1, 2), (2, 1, 2), (1, 2, 1), (2, 2, 1), (1, 2, 2), and (2, 2, 2) (see Fig. 3b). The coordinates of eight eyes of an $8 \times 8 \times 8$ mesh are (2, 2, 2), (5, 2, 2), (2, 2, 5), (5, 2, 5), (2, 5, 2), (5, 5, 2), (2, 5, 5), and (5, 5, 5).

Because the eyes of a 3-D 2^k mesh are recursively defined, the $x - y - z$ coordinate system, a fixed coordinate system, is inconvenient for the proposed broadcast algorithm (Algorithm 2). Therefore, we use a set of relative coordinate systems. In a 3-D 2^k mesh, the $u_k - v_k - w_k$ coordinate system is used, as shown in Fig. 4a. Without loss of generality, the origin of the $u_k - v_k - w_k$ coordinate system is assumed to be the eye $E_k^3(1)$ of the 3-D 2^k mesh, and the u_k, v_k, and w_k axes satisfy the right-hand screw rule, as shown in Fig. 4a. A node N at (u_k, v_k, w_k) is denoted by $N(u_k, v_k, w_k)$, where u_k, v_k, and w_k are integers. Each of the eight 3-D 2^{k-1} submeshes has its own coordinate system, $u_{k-1}(i) - v_{k-1}(i) - w_{k-1}(i)$, $i = 1, 2, ..., 8$, with $E_k^3(i)$ as its origin (see Fig. 4a).

DEFINITION 4. Denote $D_k^3(u_k, v_k, w_k)$ or $D_k^3(S)$ as the TCD of a broadcast algorithm originated from a source node $S(u_k, v_k, w_k)$ in a 3-D 2^k mesh and $MD_k^3(u_k, v_k, w_k)$ or $MD_k^3(S)$ as the minimum TCD originated from a source node $S(u_k, v_k, w_k)$ among all the possible broadcast algorithms. Obviously, $MD_k^3(u_k, v_k, w_k) = \min\{D_k^3(u_k, v_k, w_k)\}$. Denote OD_k^3 as the optimal TCD for a 3-D 2^k mesh. Obviously, $OD_k^3 = \min\{MD_k^3(u_k, v_k, w_k)\}$.

In the subsequent discussion, superscript 3, representing 3-dimensional space, will be omitted in the notation of a_k^3, E_k^3, D_k^3, MD_k^3, and OD_k^3, etc., unless there is a need to distinguish them from the ones in a 2-D mesh.

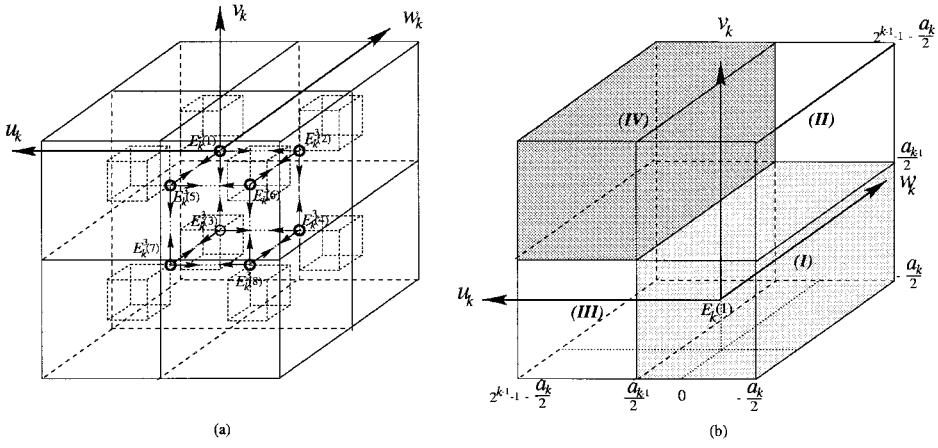

FIG. 4. (a) The $u_k - v_k - w_k$ coordinate system of a 3-D 2^k mesh. (b) Four cubic regions in the 1st submesh of a 3-D 2^k mesh.

DEFINITION 5. Denote FD as the communication distance in the first step of a broadcast, SD as the total communication distance in the second step, TD as the total communication distance in the third step, and RD as the total communication distance in the remaining steps.

For a given source node $S(u_k, v_k, w_k)$ in a 3-D mesh, different algorithms would lead to different sets of *FD*, *SD*, *TD*, and *RD*. However,

$$D_k(x, y) = FD + SD + TD + RD \tag{5}$$

can be used to calculate the *TCD* for a particular broadcast algorithm in a 3-D mesh. Note that each submesh has exactly three adjacent submeshes, one along each direction in the $u_k - v_k - w_k$ coordinate system. Without loss of generality, we assume that the first, second, and third step of a broadcast sends the message to a node in an adjacent submesh along the u_k, v_k, and w_k direction, respectively.

Figure 3a shows an example of a broadcast starting from node $S(0, 0, 0)$ in a $2 \times 2 \times 2$ mesh (3-D unit mesh). Arrows 1, 2, and 3 represent the first, second, and third steps of a broadcast, respectively. The broadcast takes three steps to complete the broadcast with each step responsible for one dimension and the number of the nodes to be delivered doubles after each step. The *TCD* of this case is calculated by $D_1(0, 0, 0) = FD + SD + TD = 2^0 + 2^1 + 2^2 = 7$. Obviously, we will obtain the same result starting from any node in this $2 \times 2 \times 2$ mesh. This means that $OD_1 = MD_1(u,v, w) = D_1(0, 0, 0) = 7$.

The following example shows a way to obtain an optimal *TCD* among all the possible source nodes in a $4 \times 4 \times 4$ mesh. We calculate the minimum *TCD* for each node in a $4 \times 4 \times 4$ mesh and place results in matrices, as shown in Fig. 5. There are four matrices in Fig. 5, each of which corresponds to a 4×4 mesh in a 2-D plane, as shown in Fig. 3b. Specifically, Figs. 5a–5d show the results of a 4×4 mesh in $y = 0$, $y = 1$, $y = 2$, and $y = 3$ planes, respectively. Within these matrices, the number at a particular position represents the minimum *TCD* if the broadcast starts from this position (node). Clearly, $OD_2 = MD_2(x, y, z) = 63$ (in the $x - y - z$ coordinate

$$\begin{pmatrix} 72 & 69 & 69 & 72 \\ 69 & 66 & 66 & 69 \\ 69 & 66 & 66 & 69 \\ 72 & 69 & 69 & 72 \end{pmatrix} \quad \begin{pmatrix} 69 & 66 & 66 & 69 \\ 66 & \underline{\mathbf{63}} & \underline{\mathbf{63}} & 66 \\ 66 & \underline{\mathbf{63}} & \underline{\mathbf{63}} & 66 \\ 69 & 66 & 66 & 69 \end{pmatrix} \quad \begin{pmatrix} 69 & 66 & 66 & 69 \\ 66 & \underline{\mathbf{63}} & \underline{\mathbf{63}} & 66 \\ 66 & \underline{\mathbf{63}} & \underline{\mathbf{63}} & 66 \\ 69 & 66 & 66 & 69 \end{pmatrix} \quad \begin{pmatrix} 72 & 69 & 69 & 72 \\ 69 & 66 & 66 & 69 \\ 69 & 66 & 66 & 69 \\ 72 & 69 & 69 & 72 \end{pmatrix}$$

(a) (b) (c) (d)

FIG. 5. The minimum *TCD*s for each node in a $4 \times 4 \times 4$ mesh (a) in $y = 0$ plane, (b) in $y = 1$ plane, (c) in $y = 2$ plane, and (d) $y = 3$ plane of Fig. 3b.

system), where (x, y, z) represents $(1, 1, 1)$, $(2, 1, 1)$, $(1, 1, 2)$, $(2, 1, 2)$, $(1, 2, 1)$, $(2, 2, 1)$, $(1, 2, 2)$, and $(2, 2, 2)$, respectively (each number with an underline in matrices). Note that these eight locations belong to eight different submeshes. In addition, these locations are exactly the locations of *eyes* of the mesh! We will show that seven of these eyes should be the destination nodes in the first three steps of a broadcast in order to achieve a minimum or an optimal broadcast.

3.2. Minimum TCD Broadcast Algorithm for a Given Source Node in a 3-D Mesh

In this section, we propose a minimum *TCD* broadcast algorithm (Algorithm 2) in a 3-D mesh and the general expression of the *TCD* obtained from this algorithm. We also show later in Theorem 3 that the *TCD* obtained from this algorithm is the minimum *TCD* for a given source node in a 3-D mesh. If the source node is an eye, the *TCD* obtained is the optimal *TCD*.

ALGORITHM 2 (Minimum *TCD* broadcast algorithm for a given source node S in a 3-D 2^k mesh).

1. Divide the given 3-D 2^k mesh into eight 3-D 2^{k-1} submeshes. Rotate the mesh, if necessary, until source node S is in the 1st submesh (where $E_k(1)$ is located), as shown in Fig. 6a.
2. In the first step, source node S sends the message to eye $E_k(2)$.
3. In the second step,

- Source node S sends the message to either eye $E_k(3)$ or $E(3)_{k-1}(5)$ (an eye in the 3rd submesh), depending on which one is closer to the source node. In other words, if source node S is in region (I) or (II) of the 1st submesh (see Fig. 4b), it sends the message to $E_k(3)$, as shown in Fig. 6a; if it is in region (III) or (IV) of the 1st submesh (see Fig. 4b), it sends the message to $E(3)_{k-1}(5)$, as shown in Fig. 6b.
- $E_k(2)$ sends the message to eye $E_k(4)$, as shown in Fig. 6a.

4. In the third step,

- Source node S sends the message to one of the eyes $E(5)_{k-1}(1)$, $E(5)_{k-1}(2)$, $E(5)_{k-1}(3)$, and $E(5)_{k-1}(4) = E_k(5)$ depending on which one is closer to the source node. In other words, if source node S is in region (IV), (II), (III), or (I) of the 1st submesh (see Fig. 4b), it sends the message to $E(5)_{k-1}(1)$, $E(5)_{k-1}(2)$, $E(5)_{k-1}(3)$, or $E(5)_{k-1}(4) = E_k(5)$, respectively.

- $E_k(2)$ sends the message to eye $E_k(6)$.
- If it is $E_k(3)$ that receives the message from source node S in the second step, $E_k(3)$ sends the message to $E_k(7)$ in this step. If it is $E(3)_{k-1}(5)$ that receives the message from source node S in the second step, $E(3)_{k-1}(5)$ sends the message to $E(7)_{k-1}(1)$ in this step.
- $E_k(4)$ sends the message to eye $E_k(8)$, as shown in Fig. 6a.

5. In the remaining steps, the eight submeshes deliver the message within their own submeshes of the next level following the above procedure (each submesh has its own eight eyes). In this way the message is delivered down to submeshes level by level until reaching the unit meshes, $2 \times 2 \times 2$ meshes, and all these unit meshes complete the broadcast within themselves in three steps, as shown in Fig. 3a.

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 2.

Source node S in Algorithm 2 can be any node in the mesh. One special case of Algorithm 2 is that source node S is an eye of the mesh, say, $E_k(1)$. In this case, source node $S(0, 0, 0) = E_k(1)$ sends the message to $E_k(2)$ in the first step of Algorithm 2, with $FD = a_k$. In the second step of Algorithm 2, source node S does not need to compare $E_k(3)$ and $E(3)_{k-1}(5)$ to determine which one is closer to it, it just sends the message to $E_k(3)$. In this step, $SD = a_k + a_k = 2a_k$. In the third step of Algorithm 2, source node S does not need to choose a destination node from $E(5)_{k-1}(1)$, $E(5)_{k-1}(2)$, $E(5)_{k-1}(3)$, and $E(5)_{k-1}(4) = E_k(5)$, it just sends the message to $E_k(5)$. Meanwhile, $E_k(3)$ sends the message to $E_k(7)$. In this step, $TD = a_k + a_k + a_k + a_k = 4a_k$. In the remaining steps, the eight submeshes deliver the message within their own submeshes following the above procedure with $RD = 8D_{k-1}(E_{k-1})$. Therefore, according to Eq. (5), $D_k(E_k)$, the TCD obtained

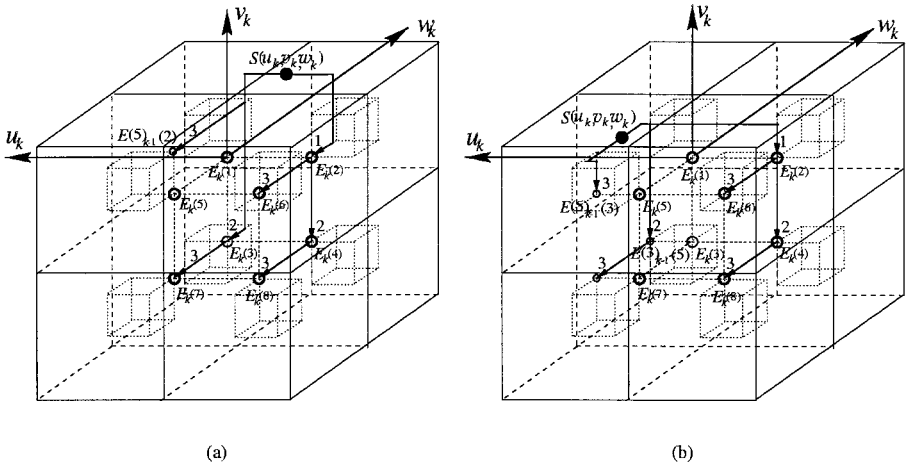

FIG. 6. The first three steps of Algorithm 2 when (a) S is in the right-half of the submesh, (b) S is in the left-half of the submesh.

from Algoritm 2 with source node $S = E_k(1)$ can be calculated by the following recursive formula:

$$D_k(E_k) = 7a_k + 8D_{k-1}(E_{k-1}). \tag{6}$$

We will show later in Theorem 4 that this $D_k(E_k)$ corresponds to the minimum TCD, $MD_k(E_k)$, and the optimal TCD, OD_k, in a 3-D mesh.

The general expression of $D_k(S)$ is very complex because $D_k(S)$ varies with the location of source node $S(u_k, v_k, w_k)$ in the 1st submesh. Based on Algorithm 2, we can divide the 1st submesh into nine different cubic regions as shown in Fig. 7, which lead to nine different expressions for $D_k(S)$. These cubic regions are marked from (i) to (ix) (see Fig. 7) and they are delimited as follows:

$$\begin{aligned}
&\text{(i)} && (-a_k/2 < u_k \leqslant 0) \wedge (-a_k/2 < v_k \leqslant 0)\\
&\text{(ii)} && (-a_k/2 < u_k \leqslant 0) \wedge (0 < v_k \leqslant a_{k-1}/2)\\
&\text{(iii)} && (-a_k/2 < u_k \leqslant 0) \wedge (a_{k-1}/2 < v_k \leqslant 2^{k-1} - 1 - a_k/2)\\
&\text{(iv)} && (0 < u_k \leqslant a_{k-1}/2) \wedge (-a_k/2 < v_k \leqslant 0)\\
&\text{(v)} && (0 < u_k \leqslant a_{k-1}/2) \wedge (0 < v_k \leqslant a_{k-1}/2)\\
&\text{(vi)} && (0 < u_k \leqslant a_{k-1}/2) \wedge (a_{k-1}/2 < v_k \leqslant 2^{k-1} - 1 - a_k/2)\\
&\text{(vii)} && (a_{k-1}/2 < u_k \leqslant 2^{k-1} - 1 - a_k/2) \wedge (-a_k/2 < v_k \leqslant 0)\\
&\text{(viii)} && (a_{k-1}/2 < u_k \leqslant 2^{k-1} - 1 - a_k/2) \wedge (0 < v_k \leqslant a_{k-1}/2)\\
&\text{(ix)} && (a_{k-1}/2 < u_k \leqslant 2^{k-1} - 1 - a_k/2)\\
&&& \wedge (a_{k-1}/2 < v_k \leqslant 2^{k-1} - 1 - a_k/2)
\end{aligned}$$

with $-a_k/2 < w_k \leqslant 2^{k-1} - 1 - a_k/2$ for all the nine cubic regions.

For example, when $S(u_k, v_k, w_k)$ is in cubic region (iii) of Fig. 7, source node S sends the message to $E_k(2)$ in the first step; source node S sends the message to $E_k(3)$ and $E_k(2)$ sends the message to $E_k(4)$ in the second step; source node S sends the message to $E(5)_{k-1}(2)$, $E_k(2)$ sends the message to $E_k(6)$, $E_k(3)$ sends the message to $E_k(7)$, and $E_k(4)$ sends the message to $E_k(8)$ in the third step, as shown in Fig. 6a. But when $S(u_k, v_k, w_k)$ is in cubic region (viii) of Fig. 7, source node S sends the message to $E_k(2)$ in the first step; source node S sends the message to $E(3)_{k-1}(5)$ and $E_k(2)$ sends the message to $E_k(4)$ in the second step; source node S sends the message to $E(5)_{k-1}(3)$, $E_k(2)$ sends the message to $E_k(6)$, $E(3)_{k-1}(5)$ sends the message to $E(7)_{k-1}(1)$, $E_k(4)$ sends the message to $E_k(8)$ in the third step, as shown in Fig. 6b. Obviously, these two cases correspond to two different expressions for $D_k(S)$. Overall, for all these nine cubic regions, we have the following theorem.

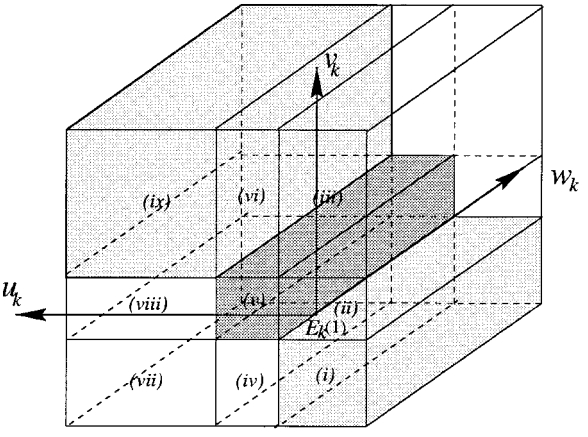

FIG. 7. Nine cubic regions in the 1st submesh of a 3-D 2^k mesh.

THEOREM 1. *Apply Algorithm* 2 *to any source node* $S(u_k, v_k, w_k)$; *the TCD obtained can be expressed as follows*:

$$D_k(S) = \begin{cases} -(u_k + v_k) + 2\,|w_k| + w_k + [7a_k + D_{k-1}(S) + 7D_{k-1}(E_{k-1})] & \text{(i)} \\ -u_k + 3v_k + 2\,|w_k| + w_k + [7a_k + D_{k-1}(S) + 7D_{k-1}(E_{k-1})] & \text{(ii)} \\ -u_k + 2v_k + |v_k - a_{k-1}| + 2\,|w_k| + w_k & \\ \quad + [7a_k + D_{k-1}(S) + 7D_{k-1}(E_{k-1})] & \text{(iii)} \\ 3u_k - v_k + 2\,|w_k| + w_k + [7a_k + D_{k-1}(S) + 7D_{k-1}(E_{k-1})] & \text{(iv)} \\ 3u_k + 3v_k + 2\,|w_k| + w_k + [7a_k + D_{k-1}(S) + 7D_{k-1}(E_{k-1})] & \text{(v)} \\ 3u_k + 2v_k + |v_k - a_{k-1}| + 2\,|w_k| + w_k & \\ \quad + [7a_k + D_{k-1}(S) + 7D_{k-1}(E_{k-1})] & \text{(vi)} \\ u_k + 2\,|u_k - a_{k-1}| - v_k + 2\,|w_k| + w_k & \\ \quad + [7a_k + D_{k-1}(S) + 7D_{k-1}(E_{k-1})] & \text{(vii)} \\ u_k + 2\,|u_k - a_{k-1}| + 3v_k + 2\,|w_k| + w_k & \\ \quad + [7a_k + D_{k-1}(S) + 7D_{k-1}(E_{k-1})] & \text{(viii)} \\ u_k + 2\,|u_k - a_{k-1}| + 2v_k + |v_k - a_{k-1}| + 2\,|w_k| + w_k & \\ \quad + [7a_k + D_{k-1}(S) + 7D_{k-1}(E_{k-1})] & \text{(ix)} \end{cases}$$

The proof of Theorem 1 is in Appendix A. Obviously, these $D_k(S)$s are too complex. To simplify them, we introduce a mapping function, denoted as $f_k(u_k, v_k, w_k)$ or simply $f_k(S)$. With this function, we can use one general expression to represent these $D_k(S)$s.

DEFINITION 6. Define function

$$f_k(u_k, v_k, w_k) = \begin{cases} 2|u_k| + u_k + 2|v_k| + v_k + 2|w_k| + w_k & \text{(I)} \\ 2|u_k| + u_k + 2|v_k| + |v_k - a_{k-1}| + 2|w_k| + w_k & \text{(II)} \\ |u_k| + 2|u_k - a_{k-1}| + 2|v_k| + v_k + 2|w_k| + w_k & \text{(III)} \\ |u_k| + 2|u_k - a_{k-1}| + 2|v_k| + |v_k - a_{k-1}| + 2|w_k| + w_k & \text{(IV)} \end{cases} \quad (7)$$

in a 3-D 2^k mesh where (I), (II), (III), and (IV) represent the lower-right, upper-right, lower-left, and upper-left cubic region of the 1st submesh, respectively, as shown in Fig. 4b, and they are delimited as follows:

(I) $(-a_k/2 < u_k \leqslant a_{k-1}/2) \wedge (-a_k/2 < v_k \leqslant a_{k-1}/2)$

(II) $(-a_k/2 < u_k \leqslant a_{k-1}/2) \wedge (a_{k-1}/2 < v_k \leqslant 2^{k-1} - 1 - a_k/2)$

(III) $(a_{k-1}/2 < u_k \leqslant 2^{k-1} - 1 - a_k/2) \wedge (-a_k/2 < v_k \leqslant a_{k-1}/2)$

(IV) $(a_{k-1}/2 < u_k \leqslant 2^{k-1} - 1 - a_k/2) \wedge (a_{k-1}/2 < v_k \leqslant 2^{k-1} - 1 - a_k/2)$

with $-a_k/2 < w_k \leqslant 2^{k-1} - 1 - a_k/2$ for all the four cubic regions.

By comparing Eq. (7) with the nine $D_k(S)$s discussed above, we have the following lemma.

LEMMA 1. *The $D_k(S)$ obtained from Algorithm 2 for a given source node $S(u_k, v_k, w_k)$ in a 3-D 2^k mesh can be calculated by a general formula*:

$$D_k(S) = f_k(u_k, v_k, w_k) + [7a_k + D_{k-1}(S) + 7D_{k-1}(E_{k-1})]. \quad (8)$$

There are two important properties of function $f_k(u_k, v_k, w_k)$:

1. Function $f_k(u_k, v_k, w_k)$ is always greater than or equal to zero, i.e.,

$$f_k(u_k, v_k, w_k) \geqslant 0. \quad (9)$$

This is because terms $2|u_k| + u_k$, $2|v_k| + v_k$, and $2|w_k| + w_k$ in Eq. (7) are always nonnegative.

2. The only condition for $f_k(u_k, v_k, w_k) = 0$ is when $S(u_k, v_k, v_k) = E_k$. In this case, Eq. (8) becomes $D_k(E_k) = 7a_k + 8D_{k-1}(E_{k-1})$, which is exactly the same as Eq. (6).

The use of function $f_k(u_k, v_k, w_k)$ not only simplifies the expression of $D_k(S)$ but also establishes the relationship between $D_k(S)$ and $D_k(E_k)$ (this is important because we have an exact formula for $D_k(E_k)$, see Eq. (13) in the next section). By comparing Eqs. (6) and (8), we can see that

$$D_k(S) - D_k(E_k) = f_k(S) + D_{k-1}(S) - D_{k-1}(E_{k-1}). \quad (10)$$

By repeatedly substituting $D_{k-1}(S)$ and $D_{k-1}(E_{k-1})$ into Eq. (10) by using Eqs. (6) and (8), respectively, we have the following lemma.

LEMMA 2. *The $D_k(S)$ obtained from Algorithm 2 for a given source node $S(u_k, v_k, w_k)$ in a 3-D 2^k mesh can be calculated in terms of $f_k(u_k, v_k, w_k)$ (or simply $f_k(S)$) and $D_k(E_k)$:*

$$D_k(S) = \sum_{i=2}^{k} f_i(S) + D_k(E_k). \tag{11}$$

Lemma 2 immediately leads to the following theorem.

THEOREM 2. *If the source node is an eye of the mesh, the TCD obtained by Algorithm 2 is the minimum one among results obtained by applying Algorithm 2 to all the possible source nodes.*

Proof. According to Eq. (9), each $f_i(S)$ in Eq. (11) is greater than or equal to zero. Therefore,

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

where the equal sign is taken only when $S(u_k, v_k, w_k) = E_k$. Thus, Theorem 2 is true. ∎

The following theorem shows that Algorithm 2 is the best possible broadcast algorithm.

THEOREM 3. *The TCD obtained from Algorithm 2 is the minimum TCD for a given source node in a 3-D mesh.*

The proof of Theorem 3 is lengthy and is placed in the Appendix B.

3.3. Optimal TCD Broadcast Algorithm for a 3-D Mesh

In the previous section, we discussed the minimum *TCD* broadcast algorithm in a 3-D mesh, which generates a minimum *TCD* for a given source node (i.e., generates an optimal *TCD* for that particular source node). In this section, we propose an optimal *TCD* broadcast algorithm (Algorithm 3) for a 3-D mesh which guarantees that the *TCD* generated by this algorithm is the optimal one among all the possible minimum *TCD*s in the mesh.

ALGORITHM 3 (Optimal *TCD* broadcast algorithm for a 3-D 2^k mesh).

Start a broadcast from an eye, say $E_k(1)$, of a 3-D mesh and follow Algorithm 2 except for the second and third steps. In the second step, source node $E_k(1)$ sends the message to eye $E_k(3)$ directly. In the third step, source node $E_k(1)$ sends the message to eye $E_k(5)$ directly and $E_k(3)$ sends the message to $E_k(7)$.

THEOREM 4. *The TCD obtained from Algorithm 3 is the optimal TCD, i.e., $OD_k = MD_k(Ek) = \min\{MD_k(S)\}$.*

Proof. Theorem 4 can be directly derived from Theorems 1 and 2. ∎

In fact, the optimal *TCD* broadcast algorithm for a 3-D 2^k mesh is just a special case of Algorithm 2 in which the broadcast originates from an eye of the mesh. This special case is discussed in the previous subsection. Therefore, Eq. (6) can be used as the recursive formula of the optimal *TCD* of a 3-D mesh, i.e.,

$$OD_k = MD_k(E_k) = 7a_k + 8MD_{k-1}(E_{k-1}), \tag{12}$$

where $MD_1(E_1) = 3$. The exact expression of OD_k is given by the following theorem.

THEOREM 5. *The optimal TCD of a* 3-*D* 2^k *mesh can be calculated by*

$$OD_k = MD_k(E_k) = \tfrac{7}{27}[2^{3k+2} - (-1)^k - 3 \times 2^k], \qquad k \geqslant 1. \tag{13}$$

For example, $OD_1 = MD_1(E_1) = 7$, $OD_2 = MD_2(E_2) = 63$, $OD_3 = MD_3(E_3) = 525$, and $OD_4 = MD_4(E_4) = 4235$, etc. The proof of Theorem 5 can be found in [1].

4. MINIMIZING *TCD* OF A TIME-STEP OPTIMAL BROADCAST IN A *d*-D MESH

In this section, we outline possible extensions of the results for 2-D meshes to d-dimensional (d-D) meshes. A $\overbrace{2^k \times 2^k \times \cdots \times 2^k}^{d}$ mesh is also called a d-D 2^k mesh or simply a d-D 2^k mesh. The definition of an eye in a d-D 2^k mesh is defined as follows:

DEFINITION 7. There are 2^d eyes in a d-D 2^k mesh, labeled as $E_k^d(i)$, $0 \leqslant i \leqslant 2^d - 1$. 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, $0 \leqslant i \leqslant 2^d - 1$. Among 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, $E_2^d(i)$. 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 2^k mesh.

DEFINITION 8. Denote $MD_k^d(E_k)$ as the minimum *TCD* for a d-D 2^k mesh to complete a broadcast from an eye.

The optimal *TCD* broadcast takes three steps to complete the broadcast in a 3-D mesh 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 can deduce that it takes d steps to complete a broadcast in a d-D unit mesh. The optimal *TCD* is

$$MD_1^d(E_1) = \sum_{i=0}^{d-1} 2^i = 2^d - 1. \tag{14}$$

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 kth 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*. 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

$$MD_k^d(E_k) = \sum_{i=0}^{d-1} 2^i a_k + 2^d MD_{k-1}^d(E_{k-1}) = (2^d - 1)\, a_k + 2^d MD_{k-1}^d(E_{k-1}), \qquad (15)$$

where $k \geqslant 2$ and $MD_1^d(E_1)$ is given by Eq. (14).

THEOREM 6. *The optimal TCD of a d-D 2^k mesh can be calculated by*

$$MD_k^d(E_k) = \frac{2^d - 1}{3(2^d + 1)(2^{d-1} - 1)} [3 \times 2^{d(k+1)-1} - (-1)^k (2^{d-1} - 1) - 2^k(2^d + 1)], \qquad (16)$$

where $k, d \geqslant 1$.

The proof of Theorem 6 is shown in Appendix C.

5. CONCLUSION

In this paper we have studied the problem of minimizing total communication distance (*TCD*) of a time-step optimal broadcast in a 3-D mesh. We have identified a set of special nodes called *eyes* in a given 3-D mesh. The divide-and-conquer approach is applied to achieve a minimum *TCD* broadcasting where a given 3-D mesh is partitioned into eight submeshes of equal size. The source node sends the broadcast message to an *eye* in each submesh. The optimization problem is then solved recursively in each submesh. Specifically, we have proposed a minimum *TCD* broadcast algorithm for a given source node in a 3-D mesh and an optimal *TCD* broadcast algorithm for a 3-D mesh. Both algorithms 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 for this given source node. If we start a broadcast from an eye of a mesh and follow the optimal *TCD* algorithm, an optimal *TCD* is obtained, which is the minimum one among *TCD*s for all the possible source nodes.

Our results can be easily extended to a 3-D torus. A torus 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 identical, 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 use the proposed optimal *TCD* algorithm for a 3-D mesh and obtain an optimal *TCD* in a 3-D torus. Another possible future work is to extend our model to a general d-dimensional mesh.

APPENDIX A

Proof of Theorem 1

The theorem can be proved by deriving expressions of $D_k(S)$ for all nine cubic regions of Fig. 7. Because the derivations for all nine cubic regions are similar, we just show detailed derivations for two regions.

First, we derive the expression of $D_k(S)$ when $S(u_k, v_k, w_k)$ is in cubic region (iii) of Fig. 7, where $-a_k/2 < u_k \leqslant 0$, $a_{k-1}/2 < v_k \leqslant 2^{k-1} - 1 - a_k/2$, and $0 < w_k \leqslant 2^{k-1} - 1 - a_k/2$. Figure 6a shows the first three steps of the broadcast. Figure 8 also shows the first three steps of the broadcast of this case but in a 2-D plane, in which Fig. 8a shows the projection of Fig. 6a along the $+w_k$ direction and Fig. 8b shows the projection of Fig. 6a along the $+u_k$ direction. In Fig. 8, only the nodes involved in the first three steps of the broadcast are drawn. Again, arrows 1, 2, and 3 represent the first, second, and third steps of the broadcast, respectively. In the first step, source node S sends the message to $E_k(2)$ with $FD = a_k + u_k + v_k + w_k$. In the second step, source node S sends the message to $E_k(3)$ with $TCD = a_k - u_k + v_k + w_k$. Meanwhile, $E_k(2)$ sends the message to $E_k(4)$ with $TCD = a_k$. Therefore, $SD = 2a_k - u_k + v_k + w_k$. In the third step, source node S sends the message to $E(5)_{k-1}(2)$ with $TCD = a_k - u_k + |v_k - a_{k-1}| + w_k$. Meanwhile, $E_k(2)$ sends the message to $E_k(6)$ with $TCD = a_k$, $E_k(3)$ sends the message to $E_k(7)$ with $TCD = a_k$, $E_k(4)$ sends the message to $E_k(8)$ with $TCD = a_k$. Therefore, $TD = 4a_k - u_k + |v_k - a_{k-1}| + w_k$. After the first three steps, each eye (seven in all except the one in the 1st submesh) has a copy of the broadcast message. This means that the *TCD* of the

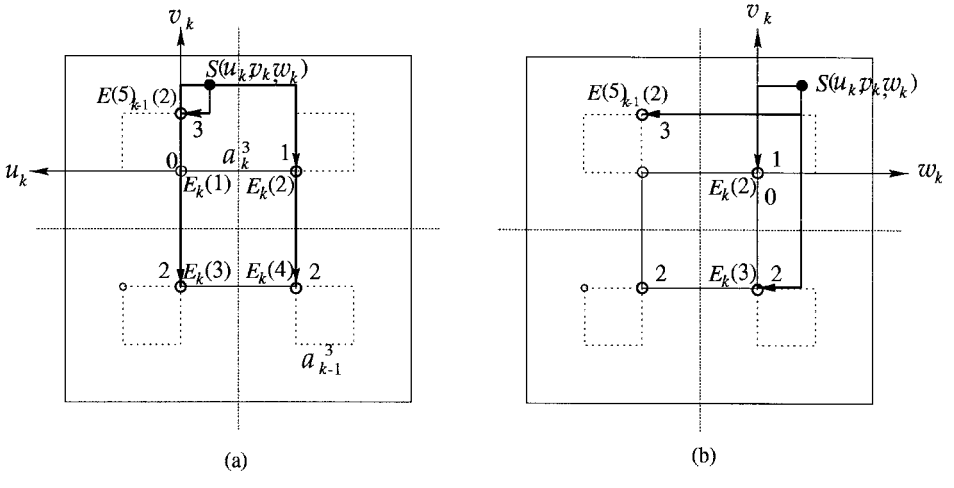

FIG. 8. The first three steps of $D_k(S)$ when $S(u_k, v_k, w_k)$ is in cubic region (iii) of the 1st submesh.

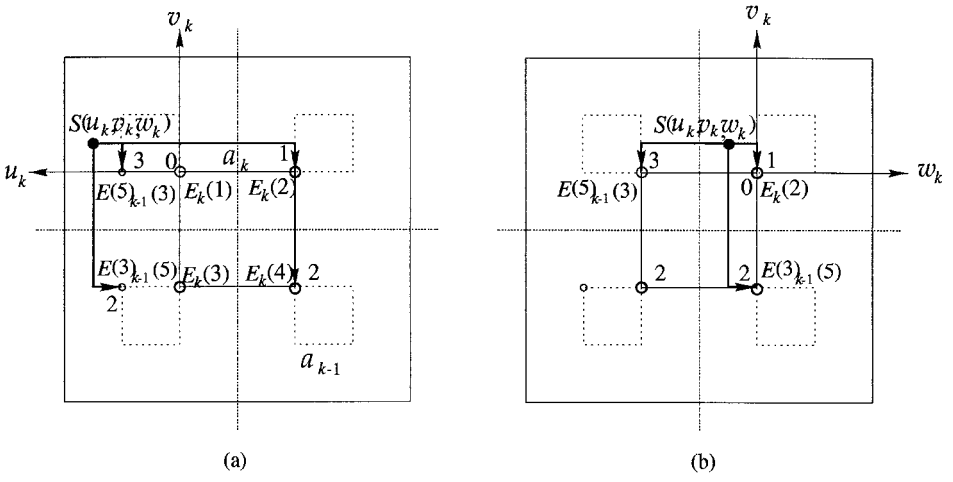

FIG. 9. The first three steps of $D_k(S)$ when $S(u_k, v_k, w_k)$ is in cubic region (viii) of the 1st submesh.

remaining steps of the broadcast can be written as $RD = D_{k-1}(S) + 7D_{k-1}(E_{k-1})$. Therefore, by using Eq. (5), we have the expression of $D_k(S)$ obtained from Algorithm 2 with source node S located in cubic region (iii) of the 1st submesh (specifically, in cubic region (iii) where $w_k \geqslant 0$),

$$D_k(S) = -u_k + 2v_k + |v_k - a_{k-1}| + 3w_k + D_{k-1}(S) + 7a_k + 7D_{k-1}(E_{k-1}).$$

Second, we derive the expression of $D_k(S)$ when $S(u_k, v_k, w_k)$ is in cubic region (viii) of Fig. 7, where $a_{k-1}/2 < u_k \leqslant 2^{k-1} - 1 - a_k/2$, $0 < v_k \leqslant a_{k-1}/2$, and $-a_k/2 < w_k \leqslant 0$. Figure 6b shows the first three steps of the broadcast. Figure 9 also shows the first three steps of the broadcast of this case but in a 2-D plane in which Fig. 9a shows the projection of Fig. 6b along the $+w_k$ direction and Fig. 9b shows the projection of Fig. 6b along the $+u_k$ direction. In Fig. 9, only the nodes involved in the first three steps of the broadcast are drawn. Again, arrows 1, 2, and 3 represent the first, second, and third steps of the broadcast, respectively. In the first step, source node S sends the message to $E_k(2)$ with $FD = a_k + u_k + v_k - w_k$. In the second step, source node S sends the message to $E(3)_{k-1}(5)$ with $TCD = a_k + |u_k - a_{k-1}| + v_k - w_k$. Meanwhile, $E_k(2)$ sends the message to $E_k(4)$ with $TCD = a_k$. Therefore, $SD = 2a_k + |u_k - a_{k-1}| + v_k - w_k$. In the third step, source node S sends the message to $E(5)_{k-1}(3)$ with $TCD = a_k + |u_k - a_{k-1}| + v_k + w_k$. Meanwhile, $E_k(2)$ sends the message to $E_k(6)$ with $TCD = a_k$, $E(3)_{k-1}(5)$ sends the message to $E(7)_{k-1}(1)$ with $TCD = a_k$, $E_k(4)$ sends the message to $E_k(8)$ with $TCD = a_k$. Therefore, $TD = 4a_k + |u_k - a_{k-1}| + v_k + w_k$. After the first three steps, each eye (seven in all except the one in the 1st submesh) has a copy of the broadcast message. This means that the TCD of the remaining steps of the broadcast can be written as $RD = D_{k-1}(S) + 7D_{k-1}(E_{k-1})$. Therefore, by using Eq. (5), we have the expression of $D_k(S)$ obtained from Algorithm 2 with source node S located in cubic region (viii) of the 1st submesh (specifically, in cubic region (viii) where $w_k \leqslant 0$),

$$D_k(S) = u_k + 2|u_k - a_{k-1}| + 3v_k - w_k + D_{k-1}(S) + 7a_k + 7D_{k-1}(E_{k-1}).$$

By treating the other seven cases in the same way, the expression of $D_k(S)$ can be derived for each case. Therefore, Theorem 1 is true. ▮

APPENDIX B

Proof of Theorem 3

We prove this theorem by mathematical induction on l in a 3-D 2^l mesh. Assume that the source node is represented by S. Theorem 3 can be proved by showing

$$D_l(S)' - D_l(S) \geqslant 0 \tag{17}$$

for any integer l, where $D_l(S)$ is the *TCD* obtained from Algorithm 2 and $D_l(S)'$ is the *TCD* obtained from an arbitrary broadcast algorithm.

- $l=1$: For $l=1$, it is a $2\times 2\times 2$ mesh (3-D unit mesh). Algorithm 2 is the only possible approach, and hence, $D_1(S)' = D_1(S) = 7$ (see Section 3). Therefore, Theorem 3 is true for $l=1$.
- $l=k-1$: Assume that Theorem 3 is true up to level $l=k-1$, i.e., $D_l(S)' - D_l(S) \geqslant 0$, $1 \leqslant l \leqslant k-1$, and $D_{k-1}(S) = MD_{k-1}(S)$. We will show that Theorem 3 is also true for level $l=k$.
- $l=k$: For $l=k$, we need to prove

$$D_k(S)' - D_k(S) \geqslant 0. \tag{18}$$

In order to prove Eq. (18), we need to determine $D_k(S)'$ and $D_k(S)$ first. For $D_k(S)'$, assuming that source node $S(u_k, v_k, w_k)$ is in the 1st submesh, the seven destination nodes of the first three steps of the broadcast are in the seven other different submeshes. Because $D_k(S)'$ is for an arbitrarily selected broadcast algorithm other than Algorithm 2, these seven destination nodes of the first three steps can be

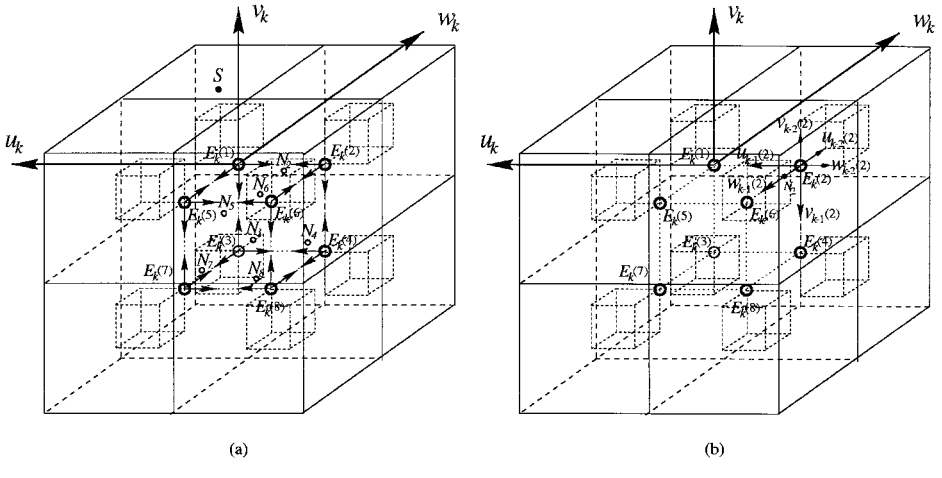

FIG. 10. (a) Calculation of $D_k(S)'$ in a 3-D 2^k mesh. (b) $f_{k-2}(N_2)$ in a 3-D 2^k mesh.

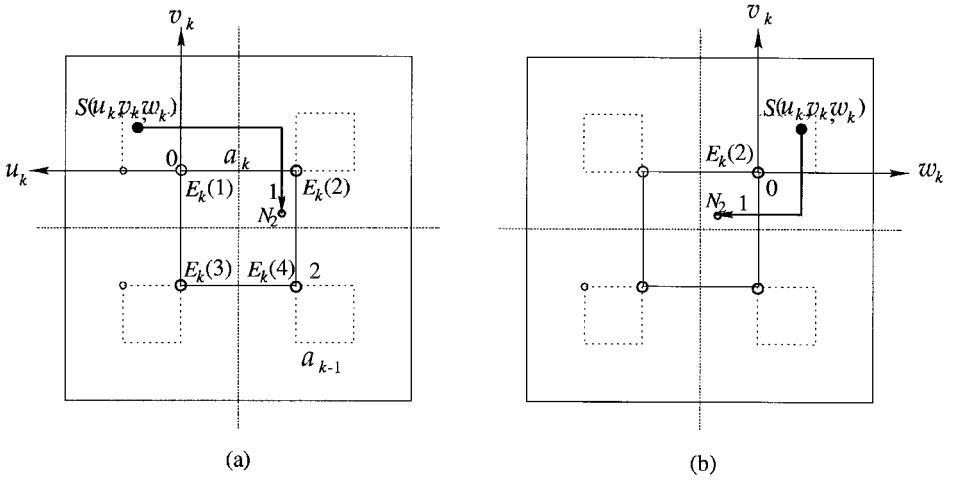

FIG. 11. The first step of broadcast of $D_k(S)'$ in a 3-D 2^k mesh.

at any locations in the corresponding submeshes. We denote these seven destination nodes of the first three steps as $N_i(u_{k-1}(i), v_{k-1}(i), w_{k-1}(i))$, $i = 2, 3, 4, ..., 8$, respectively, as shown in Figs. 10a, 11, 12, and 13. Note that the coordinates of these seven destination nodes are with respect to their own coordinate systems, not with respect to the coordinate system of the 3-D 2^k mesh. The coordinate system of each 3-D 2^{k-1} submesh is set up according to the convention in Fig. 4a.

Figure 10a shows the first three steps of $D_k(S)'$ in a 3-D 2^k mesh in a three-dimensional space. Figures 11, 12, and 13 show the first, second, and third steps of $D_k(S)'$ of a 3-D 2^k mesh in 2-D planes, respectively, in which Figs. 11a, 12a, and 13a show the projections of Fig. 10a along the $+w_k$ direction and Figs. 11b, 12b, and 13b show the projections of Fig. 10a along the $+u_k$ direction. In these figures, only the nodes involved in the corresponding steps of the broadcast are drawn.

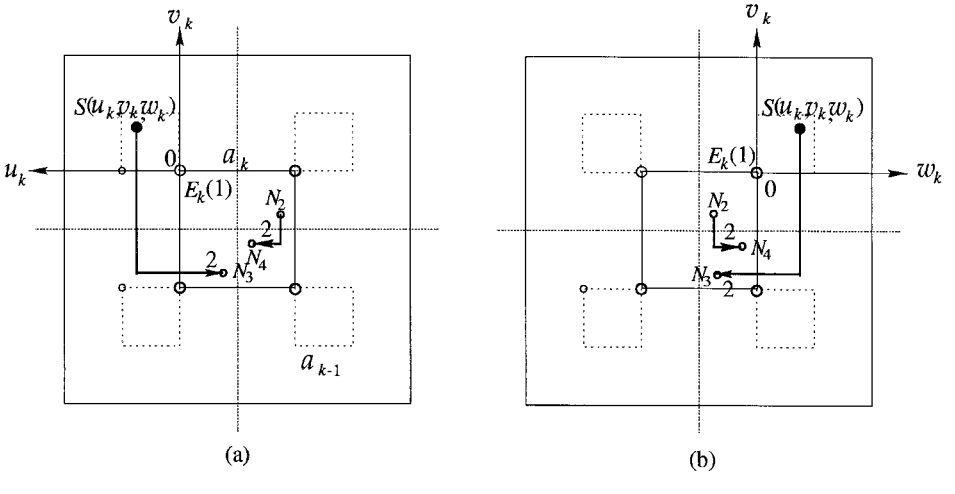

FIG. 12. The second step of broadcast of $D_k(S)'$ in a 3-D 2^k mesh.

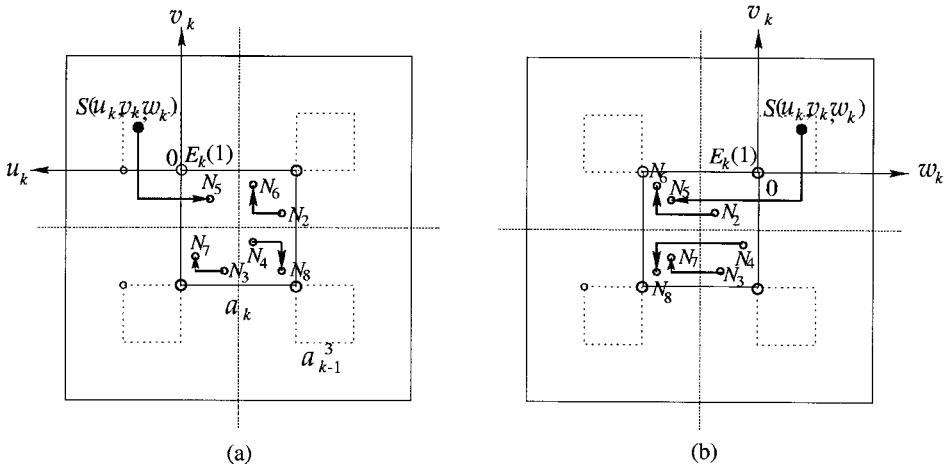

FIG. 13. The third step of broadcast of $D_k(S)'$ in a 3-D 2^k mesh.

As shown in Figs. 10a and 11, the first step of the broadcast of $D_k(S)'$ is from source node $S(u_k, v_k, w_k)$ to $N_2(u_{k-1}(2), v_{k-1}(2), w_{k-1}(2))$ with

$$FD = a_k + u_k - u_{k-1}(2) + |v_k + v_{k-1}(2)| + |w_k + w_{k-1}(2)|. \tag{19}$$

The second step of broadcast of $Dk(S)'$ is from $S(u_k, v_k, w_k)$ to $N_3(u_{k-1}(3), v_{k-1}(3), w_{k-1}(3))$ with $TCD = |u_k + u_{k-1}(3)| + a_k + v_k - v_{k-1}(3) + |w_k + w_{k-1}(3)|$, and from $N_2(u_{k-1}(2), v_{k-1}(2), w_{k-1}(2))$ to $N_4(u_{k-1}(4), v_{k-1}(4), w_{k-1}(4))$ with $TCD = |u_{k-1}(2) - w_{k-1}(4)| + a_k - v_{k-1}(2) - v_{k-1}(4) + |w_{k-1}(2) - u_{k-1}(4)|$, as shown in Figs. 10a and 12. Therefore,

$$\begin{aligned} SD = {} & |u_k + u_{k-1}(3)| + a_k + v_k - v_{k-1}(3) + |w_k + w_{k-1}(3)| \\ & + |u_{k-1}(2) - w_{k-1}(4)| + a_k - v_{k-1}(2) - v_{k-1}(4) + |w_{k-1}(2) - u_{k-1}(4)|. \end{aligned} \tag{20}$$

The third step of broadcast of $D_k(S)'$ is from $S(u_k, v_k, w_k)$ to $N_5(u_{k-1}(5), v_{k-1}(5), w_{k-1}(5))$ with $TCD = |u_k + u_{k-1}(5)| + |v_k + v_{k-1}(5)| + a_k + w_k - w_{k-1}(5)$, from $N_2(u_{k-1}(2), v_{k-1}(2), w_{k-1}(2))$ to $N_6(u_{k-1}(6), v_{k-1}(6), w_{k-1}(6))$ with $TCD = |u_{k-1}(2) - w_{k-1}(6)| + |v_{k-1}(2) - v_{k-1}(6)| + a_k + w_{k-1}(2) - u_{k-1}(6)$, from $N_3(u_{k-1}(3), v_{k-1}(3), w_{k-1}(3))$ to $N_7(u_{k-1}(7), v_{k-1}(7), w_{k-1}(7))$ with $TCD = |u_{k-1}(3) - w_{k-1}(7)| + |v_{k-1}(3) - v_{k-1}(7)| + a_k - w_{k-1}(3) - u_{k-1}(7)$, and from $N_4(u_{k-1}(4), v_{k-1}(4), w_{k-1}(4))$ to $N_8(u_{k-1}(8), v_{k-1}(8), w_{k-1}(8))$ with $TCD = |w_{k-1}(4) - u_{k-1}(8)| + |v_{k-1}(4) - v_{k-1}(8)| + a_k - u_{k-1}(4) - w_{k-1}(8)$, as shown in Figs. 10a and 13. Therefore,

$$\begin{aligned} TD = {} & |u_k + u_{k-1}(5)| + |v_k + v_{k-1}(5)| + a_k + w_k - w_{k-1}(5) \\ & + |u_{k-1}(2) - w_{k-1}(6)| + |v_{k-1}(2) - v_{k-1}(6)| + a_k + w_{k-1}(2) - u_{k-1}(6) \\ & + |u_{k-1}(3) - w_{k-1}(7)| + |v_{k-1}(3) - v_{k-1}(7)| + a_k - w_{k-1}(3) - u_{k-1}(7) \\ & + |w_{k-1}(4) - u_{k-1}(8)| + |v_{k-1}(4) - v_{k-1}(8)| + a_k - u_{k-1}(4) - w_{k-1}(8). \end{aligned} \tag{21}$$

Because Theorem 3 is assumed to be true up to level $l=k-1$, RD, the TCD of remaining steps, can be determined by

$$RD=\sum_{i=1}^{8} MD_{k-1}(N_i), \tag{22}$$

where $N_1=S(u_k, v_k, w_k)$. According to Lemma 2 and Theorems 3 and 4, we have

$$MD_k(S)=\sum_{i=2}^{k} f_i(S)+MD_k(E_k), \tag{23}$$

where the expression for $MD_k(E_k)$ is given by Eq. (13). Therefore, $MD_{k-1}(N_i)$ can be obtained by using Eq. (23), i.e.,

$$MD_{k-1}(N_i)=MD_{k-1}(E_{k-1})+\sum_{j=2}^{k-1} f_j(N_i), \tag{24}$$

where $i=1, 2, ..., 8$. By substituting Eqs. (19), (20), (21), (22), and (24) into Eq. (5), the basic formula to calculate the TCD for a particular broadcast algorithm, i.e., $D_k(S)'=FD+SD+TD+RD$, we have

$$\begin{aligned} D_k(S)'=&\sum_{i=1}^{8}\sum_{j=2}^{k-1} f_j(N_i)+8MD_{k-1}(E_{k-1})+7a_k+u_k+v_k+w_k\\ &-u_{k-1}(2)-u_{k-1}(4)-u_{k-1}(6)-u_{k-1}(7)-v_{k-1}(2)-v_{k-1}(3)\\ &-v_{k-1}(4)+w_{k-1}(2)-w_{k-1}(3)-w_{k-1}(5)-w_{k-1}(8)\\ &+|u_k+u_{k-1}(3)|+|u_k+u_{k-1}(5)|+|v_k+v_{k-1}(2)|+|v_k+v_{k-1}(5)|\\ &+|w_k+w_{k-1}(2)|+|w_k+w_{k-1}(3)|\\ &+|u_{k-1}(2)-w_{k-1}(4)|+|w_{k-1}(2)-u_{k-1}(4)|\\ &+|u_{k-1}(2)-w_{k-1}(6)|+|v_{k-1}(2)-v_{k-1}(6)|+|u_{k-1}(3)-w_{k-1}(7)|\\ &+|v_{k-1}(3)-v_{k-1}(7)|+|w_{k-1}(4)-u_{k-1}(8)|+|v_{k-1}(4)-v_{k-1}(8)|. \end{aligned} \tag{25}$$

$D_k(S)$, the TCD obtained from Algorithm 2, can be easily determined by using Eqs. (11) and (6), and hence,

$$D_k(S)=7a_k+8MD_{k-1}(E_{k-1})+\sum_{j=2}^{k} f_j(S). \tag{26}$$

Note that during the derivation of Eq. (26), we use $D_{k-1}(E_{k-1})=MD_{k-1}(E_{k-1})$ because Theorem 3 is true at level $l=k-1$.

With Eqs. (25) and (26), we are ready to prove Eq. (18). Subtracting Eq. (25) by Eq. (26), we have

$$D_k(S)' - D_k(S) = \sum_{i=0}^{8} \Delta_i, \tag{27}$$

where

$$\begin{aligned}
\Delta_0 &= |u_{k-1}(2) - w_{k-1}(4)| + |w_{k-1}(2) - u_{k-1}(4)| \\
&\quad + |u_{k-1}(2) - w_{k-1}(6)| + |v_{k-1}(2) - v_{k-1}(6)| \\
&\quad + |u_{k-1}(3) - w_{k-1}(7)| + |v_{k-1}(3) - v_{k-1}(7)| \\
&\quad + |w_{k-1}(4) - u_{k-1}(8)| + |v_{k-1}(4) - v_{k-1}(8)| \\
\Delta_1 &= u_k + v_k + w_k - f_k(S) + |u_k + u_{k-1}(3)| + |u_k + u_{k-1}(5)| \\
&\quad + |v_k + v_{k-1}(2)| + |v_k + v_{k-1}(5)| + |w_k + w_{k-1}(2)| + |w_k + w_{k-1}(3)| \qquad (28) \\
\Delta_2 &= \sum_{j=2}^{k-1} f_j(N_2) - u_{k-1}(2) - v_{k-1}(2) + w_{k-1}(2) \qquad (29) \\
\Delta_3 &= \sum_{j=2}^{k-1} f_j(N_3) - v_{k-1}(3) - w_{k-1}(3) \qquad (30) \\
\Delta_4 &= \sum_{j=2}^{k-1} f_j(N_4) - u_{k-1}(4) - v_{k-1}(4) \\
\Delta_5 &= \sum_{j=2}^{k-1} f_j(N_5) - w_{k-1}(5) \qquad (31) \\
\Delta_6 &= \sum_{j=2}^{k-1} f_j(N_6) - u_{k-1}(6) \\
\Delta_7 &= \sum_{j=2}^{k-1} f_j(N_7) - u_{k-1}(7) \\
\Delta_8 &= \sum_{j=2}^{k-1} f_j(N_8) - w_{k-1}(8).
\end{aligned}$$

In the following steps, we show that Δ_0 and Δ_i, $i = 2, 3, ..., 8$, are always nonnegative. Δ_1 is either positive or negative depending on the location of source node S and the locations of N_2, N_3, and N_5. When Δ_1 is nonnegative, Eq. (27) is automatically nonnegative, and hence, Eq. (18) is true. When Δ_1 is negative, we can show that $\Delta_1 + \Delta_2 + \Delta_3 + \Delta_5$ is still nonnegative which guarantees that Eq. (27) is nonnegative. Therefore, Eq. (18) is always true. First of all, we need to derive an important inequality to be used in the following proof. According to Eq. (7), function $f_{k-1}(N_i)$, $N_i = (u_{k-1}(i), v_{k-1}(i), w_{k-1}(i))$, $i = 1, 2, ..., 8$, can be written as:

$$f_{k-1}(N_i) = \begin{cases} 2\,|u_{k-1}(i)| + u_{k-1}(i) + 2\,|v_{k-1}(i)| \\ \qquad + v_{k-1}(i) + 2\,|w_{k-1}(i)| + w_{k-1}(i) & \text{(I)} \\ 2\,|u_{k-1}(i)| + u_{k-1}(i) + 2\,|v_{k-1}(i)| \\ \qquad + |v_{k-1}(i) - a_{k-2}| + 2\,|w_{k-1}(i)| + w_{k-1}(i) & \text{(II)} \\ |u_{k-1}(i)| + 2\,|u_{k-1}(i) - a_{k-2}| + 2\,|v_{k-1}(i)| \\ \qquad + v_{k-1}(i) + 2\,|w_{k-1}(i)| + w_{k-1}(i) & \text{(III)} \\ |u_{k-1}(i)| + 2\,|u_{k-1}(i) - a_{k-2}| + 2\,|v_{k-1}(i)| \\ \qquad + |v_{k-1}(i) - a_{k-2}| + 2\,|w_{k-1}(i)| + w_{k-1}(i), & \text{(IV)} \end{cases}$$

where (I), (II), (III), and (IV) belong to the ith submesh of a 3-D 2^k mesh. Obviously, the following inequality is always true,

$$f_{k-1}(N_i) \geqslant |u_{k-1}(i)| + |v_{k-1}(i)| + |w_{k-1}(i)|. \tag{32}$$

Now we discuss each Δ_i one by one. Note that Δ_1 is the last one to be discussed because it uses the results of other Δ_is.

Δ_0: It is obvious that $\Delta_0 \geqslant 0$.

Δ_2: According to Eq. (32), function $f_{k-1}(N_2(u_{k-1}(2), v_{k-1}(2), w_{k-1}(2)))$ can be written as:

$$f_{k-1}(N_2) \geqslant |u_{k-1}(2)| + |v_{k-1}(2)| + |w_{k-1}(2)|.$$

Therefore, Δ_2 can be rewritten as

$$\Delta_2 \geqslant \sum_{j=2}^{k-2} f_j(N_2) + (|u_{k-1}(2)| - u_{k-1}(2)) + (|v_{k-1}(2)| - v_{k-1}(2)) + (|w_{k-1}(2)| + w_{k-1}(2))$$

by substituting $f_{k-1}(N_2)$ into the Δ_2 expression. Because $|u_{k-1}(2)| - u_{k-1}(2) \geqslant 0$, $|v_{k-1}(2)| - v_{k-1}(2) \geqslant 0$, $|w_{k-1}(2)| + w_{k-1}(2) \geqslant 0$, and $\sum_{j=2}^{k-2} f_j(N_2) \geqslant 0$ (according to Eq. (9)), we immediately have $\Delta_2 \geqslant 0$.

Δ_3: According to Eq. (32), function $f_{k-1}(N_3(u_{k-1}(3), v_{k-1}(3), w_{k-1}(3)))$ can be written as:

$$f_{k-1}(N_3) \geqslant |u_{k-1}(3)| + |v_{k-1}(3)| + |w_{k-1}(3)|.$$

Therefore, Δ_3 can be rewritten as

$$\Delta_3 \geqslant \sum_{j=2}^{k-2} f_j(N_3) + |u_{k-1}(3)| + (|v_{k-1}(3)| - v_{k-1}(3)) + (|w_{k-1}(3)| - w_{k-1}(3))$$

by substituting $f_{k-1}(N_3)$ into the Δ_3 expression. Because $|u_{k-1}(3)| \geqslant 0$, $|v_{k-1}(3)| - v_{k-1}(3) \geqslant 0$, $|w_{k-1}(3)| - w_{k-1}(3) \geqslant 0$, and $\sum_{j=2}^{k-2} f_j(N_3) \geqslant 0$, we immediately have $\Delta_3 \geqslant 0$.

Δ_4: According to Eq. (32), function $f_{k-1}(N_4(u_{k-1}(4), v_{k-1}(4), w_{k-1}(4)))$ can be written as:

$$f_{k-1}(N_4) \geqslant |u_{k-1}(4)| + |v_{k-1}(4)| + |w_{k-1}(4)|.$$

Therefore, Δ_4 can be rewritten as

$$\Delta_4 \geqslant \sum_{j=2}^{k-2} f_j(N_4) + (|u_{k-1}(4)| - u_{k-1}(4)) + (|v_{k-1}(4)| - v_{k-1}(4)) + |w_{k-1}(4)|$$

by substituting $f_{k-1}(N_4)$ into the Δ_4 expression. Because $|u_{k-1}(4)| - u_{k-1}(4) \geqslant 0$, $|v_{k-1}(4)| - v_{k-1}(4) \geqslant 0$, $|w_{k-1}(4)| \geqslant 0$, and $\sum_{j=2}^{k-2} f_j(N_4) \geqslant 0$, we immediately have $\Delta_4 \geqslant 0$.

Δ_5: According to Eq. (32), function $f_{k-1}(N_5(u_{k-1}(5), v_{k-1}(5), w_{k-1}(5)))$ can be written as:

$$f_{k-1}(N_5) \geqslant |u_{k-1}(5)| + |v_{k-1}(5)| + |w_{k-1}(5)|.$$

Therefore, Δ_5 can be rewritten as

$$\Delta_5 \geqslant \sum_{j=2}^{k-2} f_j(N_5) + |u_{k-1}(5)| + |v_{k-1}(5)| + (|w_{k-1}(5)| - w_{k-1}(5))$$

by substituting $f_{k-1}(N_5)$ into the Δ_5 expression. Because $|u_{k-1}(5)| \geqslant 0$, $|v_{k-1}(5)| \geqslant 0$, $|w_{k-1}(5)| - w_{k-1}(5) \geqslant 0$, and $\sum_{j=2}^{k-2} f_j(N_5) \geqslant 0$, we immediately have $\Delta_5 \geqslant 0$.

Δ_6: According to Eq. (32), function $f_{k-1}(N_6(u_{k-1}(6), v_{k-1}(6), w_{k-1}(6)))$ can be written as:

$$f_{k-1}(N_6) \geqslant |u_{k-1}(6)| + |v_{k-1}(6)| + |w_{k-1}(6)|.$$

Therefore, Δ_6 can be rewritten as

$$\Delta_6 \geqslant \sum_{j=2}^{k-2} f_j(N_6) + (|u_{k-1}(6)| - u_{k-1}(6)) + |v_{k-1}(6)| + |w_{k-1}(6)|$$

by substituting $f_{k-1}(N_6)$ into the Δ_6 expression. Because $|u_{k-1}(6)| - u_{k-1}(6) \geqslant 0$, $|v_{k-1}(6)| \geqslant 0$, $|w_{k-1}(6)| \geqslant 0$, and $\sum_{j=2}^{k-2} f_j(N_6) \geqslant 0$, we immediately have $\Delta_6 \geqslant 0$.

Δ_7: According to Eq. (32), function $f_{k-1}(N_7(u_{k-1}(7), v_{k-1}(7), w_{k-1}(7)))$ can be written as:

$$f_{k-1}(N_7) \geqslant |u_{k-1}(7)| + |v_{k-1}(7)| + |w_{k-1}(7)|.$$

Therefore, Δ_7 can be rewritten as

$$\Delta_7 \geqslant \sum_{j=2}^{k-2} f_j(N_7) + (|u_{k-1}(7)| - u_{k-1}(7)) + |v_{k-1}(7)| + |w_{k-1}(7)|$$

by substituting $f_{k-1}(N_7)$ into the Δ_7 expression. Because $|u_{k-1}(7)| - u_{k-1}(7) \geqslant 0$, $|v_{k-1}(7)| \geqslant 0$, $|w_{k-1}(7)| \geqslant 0$, and $\sum_{j=2}^{k-2} f_j(N_7) \geqslant 0$, we immediately have $\Delta_7 \geqslant 0$.

Δ_8: According to Eq. (32), function $f_{k-1}(N_8(u_{k-1}(8), v_{k-1}(8), w_{k-1}(8)))$ can be written as:

$$f_{k-1}(N_8) \geqslant |u_{k-1}(8)| + |v_{k-1}(8)| + |w_{k-1}(8)|.$$

Therefore, Δ_8 can be rewritten as

$$\Delta_8 \geqslant \sum_{j=2}^{k-2} f_j(N_8) + |u_{k-1}(8)| + |v_{k-1}(8)| + (|w_{k-1}(8)| - w_{k-1}(8))$$

by substituting $f_{k-1}(N_8)$ into the Δ_8 expression. Because $|u_{k-1}(8)| \geqslant 0$, $|v_{k-1}(8)| \geqslant 0$, $|w_{k-1}(8)| - w_{k-1}(8) \geqslant 0$, and $\sum_{j=2}^{k-2} f_j(N_8) \geqslant 0$, we immediately have $\Delta_8 \geqslant 0$.

Δ_1: Δ_1 in Eq. (27) is rather complex because it can be either positive or negative depending on the location of S and the locations of N_2, N_3, and N_5. In the following, we show that

$$\Delta_1 + \Delta_2 + \Delta_3 + \Delta_5 \geqslant 0 \tag{33}$$

even when Δ_1 is negative.

When $u_k, v_k, w_k \leqslant 0$, according to Eq. (7), $f_k(S)$ becomes

$$f_k(S) = |u_k| + |v_k| + |w_k|.$$

Therefore, Δ_1 (see Eq. (28)) becomes

$$\begin{aligned}\Delta_1 = &-2|u_k| - 2|v_k| - 2|w_k| + |u_k + u_{k-1}(3)| + |u_k + u_{k-1}(5)| \\ &+ |v_k + v_{k-1}(2)| + |v_k + v_{k-1}(5)| + |w_k + w_{k-1}(2)| + |w_k + w_{k-1}(3)|.\end{aligned} \tag{34}$$

In Eq. (34), there are three negative terms $-2|u_k|$, $-2|v_k|$, and $-2|w_k|$. If $u_{k-1}(3)$, $u_{k-1}(5)$, $v_{k-1}(2)$, $v_{k-1}(5)$, $w_{k-1}(2)$, $w_{k-1}(3)$ in Eq. (34) are all negative, Eq. (34) becomes

$$\begin{aligned}\Delta_1 = &-2|u_k| - 2|v_k| - 2|w_k| + |u_k| + |u_{k-1}(3)| + |u_k| + |u_{k-1}(5)| \\ &+ |v_k| + |v_{k-1}(2)| + |v_k| + |v_{k-1}(5)| + |w_k| + |w_{k-1}(2)| + |w_k| + |w_{k-1}(3)|.\end{aligned}$$

The three negative terms are cancelled out. Therefore, $\Delta_1 \geqslant 0$. The worst case of Eq. (34) occurs when all the terms, $u_{k-1}(3)$, $u_{k-1}(5)$, $v_{k-1}(2)$, $v_{k-1}(5)$, $w_{k-1}(2)$, and $w_{k-1}(3)$, in Eq. (34) are positive. In this case, Eq. (34) becomes

$$\begin{aligned}\Delta_1 = &-2|u_k| - 2|v_k| - 2|w_k| + |u_k| - |u_{k-1}(3)| + |u_k| - |u_{k-1}(5)| \\ &+ |v_k| - |v_{k-1}(2)| + |v_k| - |v_{k-1}(5)| + |w_k| \\ &- |w_{k-1}(2)| + |w_k| - |w_{k-1}(3)|,\end{aligned}$$

i.e.,

$$\Delta_1 = -|v_{k-1}(2)| - |w_{k-1}(2)| - |u_{k-1}(3)| - |w_{k-1}(3)| - |u_{k-1}(5)| - |v_{k-1}(5)|. \tag{35}$$

Δ_1 is negative now. However, all these six negative terms in Eq. (35) can be cancelled out in $\Delta_1 + \Delta_2 + \Delta_3 + \Delta_5$. Because both $v_{k-1}(2)$ and $w_{k-1}(2)$ are positive, according to Eq. (7), $f_{k-1}(N_2)$ satisfies

$$f_{k-1}(N_2) \geqslant |u_{k-1}(2)| + 2|v_{k-1}(2)| + 3|w_{k-1}(2)|. \tag{36}$$

Substituting Eq. (36) into Eq. (29), Δ_2 can be rewritten as

$$\Delta_2 \geqslant \sum_{j=2}^{k-2} f_j(N_2) + |v_{k-1}(2)| + 4|w_{k-1}(2)|. \tag{37}$$

Therefore, terms $|v_{k-1}(2)|$ and $|w_{k-1}(2)|$ in Eq. (37) can cancel out two negative terms, $-|v_{k-1}(2)|$ and $-|w_{k-1}(2)|$, in Eq. (35). The four other negative terms in Eq. (35) can be cancelled out in a similar way. Because both $u_{k-1}(3)$ and $w_{k-1}(3)$ are positive, according to Eq. (7), $f_{k-1}(N_3)$ satisfies

$$f_{k-1}(N_3) \geqslant |u_{k-1}(3)| + |v_{k-1}(3)| + 3|w_{k-1}(3)|.$$

Substituting this inequality into Eq. (30), Δ_3 can be rewritten as

$$\Delta_3 \geqslant \sum_{j=2}^{k-2} f_j(N_3) + |u_{k-1}(3)| + 2|w_{k-1}(3)|.$$

Therefore, terms $|u_{k-1}(3)|$ and $|w_{k-1}(3)|$ in this inequality can cancel out two negative terms, $-|u_{k-1}(3)|$ and $-|w_{k-1}(3)|$, in Eq. (35). Because both $u_{k-1}(5)$ and $v_{k-1}(5)$ are positive, according to Eq. (7), $f_{k-1}(N_5)$ satisfies

$$f_{k-1}(N_5) \geqslant |u_{k-1}(5)| + 2|v_{k-1}(5)| + |w_{k-1}(5)|.$$

Substituting this inequality into Eq. (31), Δ_5 can be rewritten as

$$\Delta_5 \geqslant \sum_{j=2}^{k-2} f_j(N_5) + |u_{k-1}(5)| + 2|v_{k-1}(5)|.$$

Therefore, terms $|u_{k-1}(5)|$ and $|v_{k-1}(5)|$ in the above inequality can cancel out two negative terms, $-|u_{k-1}(5)|$ and $-|v_{k-1}(5)|$, in Eq. (35). Therefore, all six negative

terms in Eq. (35) are cancelled out by $\Delta_2+\Delta_3+\Delta_5$. This means Eq. (33) is true when $u_k, v_k, w_k \leqslant 0$.

When $u_k, v_k, w_k > 0$, according to Eq. (7), the maximum value of $f_k(S)$ (which corresponds to the worst case of Δ_1) becomes

$$f_k(S) = 3\,|u_k| + 3\,|v_k| + 3\,|w_k|.$$

Therefore, Δ_1 (see Eq. (28)) becomes

$$\begin{aligned}\Delta_1 = &-2\,|u_k| - 2\,|v_k| - 2\,|w_k| + |u_k + u_{k-1}(3)| + |u_k + u_{k-1}(5)| \\ &+ |v_k + v_{k-1}(2)| + |v_k + v_{k-1}(5)| + |w_k + w_{k-1}(2)| + |w_k + w_{k-1}(3)|. \end{aligned} \tag{38}$$

In Eq. (38), there are also three negative terms $-2\,|u_k|$, $-2\,|v_k|$, and $-2\,|w_k|$. When $u_{k-1}(3)$, $u_{k-1}(5)$, $v_{k-1}(2)$, $v_{k-1}(5)$, $w_{k-1}(2)$, $w_{k-1}(3)$ in Eq. (38) are all positive, Eq. (38) becomes

$$\begin{aligned}\Delta_1 = &-2\,|u_k| - 2\,|v_k| - 2\,|w_k| + |u_k| + |u_{k-1}(3)| + |u_k| + |u_{k-1}(5)| \\ &+ |v_k| + |v_{k-1}(2)| + |v_k| + |v_{k-1}(5)| + |w_k| + |w_{k-1}(2)| + |w_k| + |w_{k-1}(3)|.\end{aligned}$$

The three negative terms are cancelled out. Therefore, $\Delta_1 \geqslant 0$. The worst case of Eq. (38) occurs when all the terms, $u_{k-1}(3)$, $u_{k-1}(5)$, $v_{k-1}(2)$, $v_{k-1}(5)$, $w_{k-1}(2)$, and $w_{k-1}(3)$, in Eq. (38) are negative. In this case, Eq. (38) becomes

$$\begin{aligned}\Delta_1 = &-2\,|u_k| - 2\,|v_k| - 2\,|w_k| + |u_k| - |u_{k-1}(3)| + |u_k| - |u_{k-1}(5)| \\ &+ |v_k| - |v_{k-1}(2)| + |v_k| - |v_{k-1}(5)| + |w_k| - |w_{k-1}(2)| + |w_k| - |w_{k-1}(3)|,\end{aligned}$$

i.e.,

$$\Delta_1 = -|v_{k-1}(2)| - |w_{k-1}(2)| - |u_{k-1}(3)| - |w_{k-1}(3)| - |u_{k-1}(5)| - |v_{k-1}(5)|. \tag{39}$$

Δ_1 is negative now. We can still use six positive terms, $|v_{k-1}(2)|$, $|w_{k-1}(2)|$, $|u_{k-1}(3)|$, $|w_{k-1}(3)|$, $|u_{k-1}(5)|$, and $|v_{k-1}(5)|$, from $\Delta_2+\Delta_3+\Delta_5$ to cancel out all these six negative terms in Eq. (39). Because both $v_{k-1}(2)$ and $w_{k-1}(2)$ are negative, according to Eq. (7), $f_{k-1}(N_2)$ satisfies

$$f_{k-1}(N_2) \geqslant |u_{k-1}(2)| + |v_{k-1}(2)| + |w_{k-1}(2)|. \tag{40}$$

Substituting Eq. (40) into Eq. (29), Δ_2 can be rewritten as

$$\Delta_2 \geqslant \sum_{j=2}^{k-2} f_j(N_2) + 2\,|u_{k-1}(2)| + 2\,|v_{k-1}(2)|. \tag{41}$$

Now only one term $|v_{k-1}(2)|$ can be used to cancel out $-|v_{k-1}(2)|$ in Eq. (39). We still need term $|w_{k-1}(2)|$ to cancel out $-|w_{k-1}(2)|$ in Eq. (39). Actually, $|w_{k-1}(2)|$ can be found in $f_{k-2}(N_2)$ in Eq. (41). Because

$$f_{k-2}(N_2) \geqslant |u_{k-2}(2)| + |v_{k-2}(2)| + |w_{k-2}(2)|,$$

and $|u_{k-2}(2)| = |w_{k-1}(2)|$, as shown in Fig. 10b, we have $f_{k-2}(N_2) \geqslant |w_{k-1}(2)|$. Substituting this result into Eq. (41), we have

$$\Delta_2 \geqslant \sum_{j=2}^{k-3} f_j(N_2) + 2\,|u_{k-1}(2)| + 2\,|v_{k-1}(2)| + |w_{k-1}(2)|. \tag{42}$$

Therefore, terms $|v_{k-1}(2)|$ and $|w_{k-1}(2)|$ in Eq. (42) can cancel out the two negative terms, $-|v_{k-1}(2)|$ and $-|w_{k-1}(2)|$, in Eq. (39). The four other negative terms in Eq. (39) can be cancelled out in a similar way. Because both $u_{k-1}(3)$ and $w_{k-1}(3)$ are positive, according to Eq. (7), $f_{k-1}(N_3)$ satisfies

$$f_{k-1}(N_3) \geqslant |u_{k-1}(3)| + |v_{k-1}(3)| + |w_{k-1}(3)|.$$

Substituting this inequality into Eq. (30), Δ_3 can be rewritten as

$$\Delta_3 \geqslant \sum_{j=2}^{k-2} f_j(N_3) + |u_{k-1}(3)| + |v_{k-1}(3)| + 2\,|w_{k-1}(3)|.$$

Therefore, terms $|u_{k-1}(3)|$ and $|w_{k-1}(3)|$, in this inequality can cancel out two negative terms, $-|u_{k-1}(3)|$ and $-|w_{k-1}(3)|$, in Eq. (39). Because both $u_{k-1}(5)$ and $v_{k-1}(5)$ are positive, according to Eq. (7), $f_{k-1}(N_5)$ satisfies

$$f_{k-1}(N_5) \geqslant |u_{k-1}(5)| + |v_{k-1}(5)| + |w_{k-1}(5)|.$$

Substituting this inequality into Eq. (31), Δ_5 can be rewritten as

$$\Delta_5 \geqslant \sum_{j=2}^{k-2} f_j(N_5) + |u_{k-1}(5)| + |v_{k-1}(5)| + 2\,|w_{k-1}(5)|.$$

Therefore, terms $|u_{k-1}(5)|$ and $|v_{k-1}(5)|$, in the above inequality can cancel out two negative terms, $-|u_{k-1}(5)|$ and $-|v_{k-1}(5)|$, in Eq. (39). Hence, all six negative terms in Eq. (39) are cancelled out by $\Delta_2 + \Delta_3 + \Delta_5$. This means Eq. (33) is true.

In summary, we have shown that Δ_0, and Δ_i, $i = 2, 3, 4, ..., 8$ are always nonnegative. Δ_1 is either positive or negative. When Δ_1 is nonnegative, Eq. (27) is automatically nonnegative, and hence, Eq. (18) is true. When Δ_1 is negative, $\Delta_1 + \Delta_2 + \Delta_3 + \Delta_5$ is still nonnegative which guarantees that Eq. (27) is nonnegative, and hence, Eq. (18) is correct. Therefore, Eq. (18) is always true, which means that Eq. (17) is always true for $l = k$. Therefore, Eq. (17) is true for all l, i.e., Theorem 3 is true. ▮

APPENDIX C

Proof of Theorem 6

Proof. Equation (15) can be written as

$$MD_k^d(E_k) = (2^d - 1)[a_k + (2^d)^1 a_{k-1} + (2^d)^2 a_{k-2} + (2^d)^3 a_{k-3} + \cdots + (2^d)^{k-2} a_2] + (2^d)^{k-1} MD_1^d(E_1), \tag{43}$$

through substitution. Using Eq. (1), the first term of Eq. (43) can be written as

$$\begin{aligned} & a_k + (2^d)^1 a_{k-1} + (2^d)^2 a_{k-2} + (2^d)^3 a_{k-3} + \cdots + (2^d)^{k-2} a_2 \\ &= \frac{1}{3}\{[(2^d)^0 2^{k-0} + (2^d)^1 2^{k-1} + (2^d)^2 2^{k-2} + (2^d)^3 2^{k-3} + \cdots + (2^d)^{k-2} 2^2] \\ &\quad - [(2^d)^0 (-1)^{k-0} + (2^d)^1 (-1)^{k-1} + (2^d)^2 (-1)^{k-2} \\ &\quad + (2^d)^3 (-1)^{k-3} + \cdots + (2^d)^{k-2} (-1)^2]\} \\ &= \frac{1}{3}\{2^k[(2^{d-1})^0 + (2^{d-1})^1 + (2^{d-1})^2 + (2^{d-1})^3 + \cdots + (2^{d-1})^{k-2}] \\ &\quad - (-1)^k [(-2^d)^0 + (-2^d)^1 + (-2^d)^2 + (-2^d)^3 + \cdots + (-2^d)^{k-2}]\} \\ &= \frac{1}{3}\left[2^k \times \frac{1 - (2^{d-1})^{k-1}}{1 - 2^{d-1}} - (-1)^k \times \frac{1 + (-1)^k (2^d)^{k-1}}{1 + 2^d}\right]. \end{aligned} \tag{44}$$

Therefore, by substituting Eq. (44) and $MD_1^d(E_1) = 2^d - 1$ into Eq. (43), we get Eq. (16). ∎

REFERENCES

1. S. Cang and J. Wu, Minimizing total communication distance of a time-step optimal broadcast in mesh networks, *in* "Proceedings of the First Merged IPPS/SPDP 1998, April 1998," pp. 10–17.
2. W. J. Dally, The J-machine: system support for actors, *in* "Actors: Knowledge-based Concurrent Computing" (Hewitt and Agha, Eds.), MIT Press, Cambridge, MA, 1989.
3. J. Duato, S. Yalmanchili, and L. M. Ni, "Interconnection Networks: An Engineering Approach," IEEE Computer Society, Los Alamitos, CA, 1997.
4. S. L. Johnsson and C. T. Ho, Optimal broadcasting and personalized communication in hypercubes, *IEEE Trans. Comput.* **38**, 9 (September 1989), 1249–1268.
5. R. K. Koeninger, M. Furtney, and M. Walker, A shared memory MPP from Cray research, *Digital Tech. J.* **6**, 2 (Spring 1994), 8–21.
6. S. L. Lamport, R. Shostak, and M. Pease, The byzantine generals problems, *ACM Trans. Progr. Languages Systems* (June 1992), 632–639.
7. S. L. Lillevik, The Touchstone 30 gigaflop DELTA prototype, *in* "Proc. of the 6th Distributed Memory Computing Conference, 1991," pp. 671–677.
8. X. Lin and L. M. Ni, Multicast communication in multicomputers networks, *IEEE Trans. Parallel Distrib. Systems* **4**, 10 (October 1993), 1105–1117.
9. Message Passing Interface Forum, "MPI: A Message-Passing Interface Standard," March 1994.

10. P. A. Nelson and L. Snyder, Programming paradigms for nonshared memory parallel computers, *in* "The Characteristics of Parallel Algorithms" (L. H. Jamieson *et al*., Eds.), Vol. I, pp. 346–354, MIT Press, Cambridge, MA, 1987.
11. D. K. Panda, Issues in designing efficient and practical algorithms for collective communication on wormhole routed systems, *in* "Proc. of the 1995 ICPP Workshop on Challenges for Parallel Processing, August 1995," pp. 8–15.
12. J. Y. L. Park, H. A. Choi, N. Nupairoj, and L. M. Ni, Construction of optimal multicast trees based on the parameterized communication model, *in* "Proc. of the International Conference on Parallel Processing, August 1996," pp. 180–187.
13. P. Ramanathan, K. G. Shin, and R. W. Butler, Fault-tolerant clock synchronization in distributed systems, *Computer* **23**, 10 (October 1990), 33–42.
14. C. L. Seitz, The architecture and programming of the Ametek Series 2010 multicomputer, *in* "Proc. of the 3rd Conference on Hypercube Concurrent Computers and Applications, January 1988," pp. I33–I36.
15. Y. J. Suh and S. Yalamanchili, All-to-all communication with minimum start-up costs in 2D and 3D tori, *IEEE Trans. Parallel Distrib. Systems* **9**, 5 (May 1998), 442–458.
16. N. S. Sundar, D. N. Jayasimha, D. K. Panda, and P. Sadayappan, Complete exchange in 2D meshes, *in* "Proc. of Scalable High Performance Computing Conference, 1994," pp. 406–413.
17. Y. J. Tsai and P. K. McKinley, A broadcasting algorithm for all-port wormhole-routed torus networks, *IEEE Trans. Parallel Distrib. Systems* **7**, 8 (Aug. 1996), 947–952.
18. S. Y. Wang and Y. C. Tseng, Algebraic foundations and broadcasting algorithms for wormhole-routed all-port tori, *in* "Proc. of International Conference on Parallel Processing," pp. 543–550.
19. I. Wojciechowska, Line broadcast in grid graphs, *in* "The 28th Southeastern International Conference on Combinatorics Graph Theory Computing, March 1997." [abstract only]