ELSEVIER

Journal of Systems Architecture 42 (1996) 367–380

JOURNAL OF SYSTEMS ARCHITECTURE

Optimal broadcasting in hypercubes with link faults using limited global information

Jie Wu [1]

Department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, FL 33431, USA

Received 1 September 1995; revised 18 December 1995; accepted 20 March 1996

Abstract

We propose a fault-tolerant broadcasting algorithm for hypercubes with link faults. This algorithm is based on an extended spanning binomial tree structure that still keeps the simplicity of conventional binomial-tree-based broadcasting. In addition, it is optimal in the sense that exactly n steps are required to complete a broadcast in an n-dimensional injured hypercube with up to $n-2$ faulty links. We also show that $n-2$ is the maximum number of faulty links that can be tolerated in any optimal broadcast scheme in an n-dimensional hypercube. To implement the proposed algorithm each node keeps information of nearby faulty links in terms of addresses for those faulty adjacent m-subcubes that contain at least $m-1$ faulty links.

Keywords: Binomial tree; Broadcasting; Fault tolerance; Hypercubes

1. Introduction

The hypercube structure [12,4] is one of the most popular message-passing architectures, and several multicomputer configurations have been prototyped or marketed [5,13]. Broadcasting [6] concerns transmitting a data set from one node to all the other nodes in a network. Broadcasting is an important operation frequently used in a variety of linear algorithms, database queries, and linear programming algorithms. The basic broadcasting algorithm was introduced in [14] based on the binomial tree structure. When a component (links or nodes) in the hypercube fails during a long computation, which is not uncommon, hours of computation will be wasted if no proper fault-tolerant mechanism exists. Therefore, there is a need for fault-tolerant broadcasting dealing with successful broadcasting in the presence of faulty components. In general, fault-tolerant broadcasting can be classified based on (1) the way each destination receives the broadcasting data, (2) the amount of information kept at each node, (3) the

[1] Email: jie@cse.fau.edu.

PII S1383-7621(96)00023-9

type of faulty components, and (4) the number of faulty components.

There are in general two ways that each destination receives the broadcast data: (1) Each node might receive more than one copy, and the corresponding algorithm is called *redundant broadcasting* [11]. Normally, in a redundant broadcasting the source node simultaneously sends copies of broadcasting data to its neighbors. This approach has its merit of simplicity and it doesn't require backtracking during the broadcasting. The flaw with this approach is the extra network traffic. It is clear that redundant broadcasting is not necessary in the absence of faulty components. (2) Each node can only receive one copy of broadcast data. Therefore, broadcasting algorithms should be designed such that the broadcast data is sent to each node once and only once. Algorithms of this type are called *nonredundant broadcast algorithms*. In a broadcasting process, the amount of faulty component information kept at each node can be classified as local, limited global, and global. Local information contains only adjacent faulty components. Limited global information contains the distribution of faulty components in the neighborhood. Global information contains the distribution of all the faulty components. There are two types of faulty components: faulty link and faulty node. The number of faulty components can be either bounded or unbounded.

Among the approaches based on local network information. Al-Dhelaan and Bose [1] proposed a binomial-tree-based broadcasting for limited link and node faults. This approach was enhanced by Wu and Fernandez [17] which guarantees time-step optimal. Li and Wu [8] proposed a general broadcasting scheme with local network information that can tolerant any number of faults. However, backtracking is required and network information (faulty component information) has to be incorporated as a queue into the broadcast data. Broadcasting schemes based on global information normally use routing tables [9] to keep global information. By taking the advantage of hypercube topology, Wu [16] proposed an efficient broadcasting using global information. The type of faults under consideration is link faults and the number of faults is limited to the dimension of hypercube minus one. Baghavendra [10] studied a broadcasting approach based on the concept of free-dimension that also uses global information.

The fault-tolerant broadcasting based on local information normally requires a routing history as part of the message to be broadcast in order to reach each node once and only once. While the fault-tolerant broadcasting based on global information, although it has its merit of simplicity, requires a process which collects global information. The broadcasting based on limited global information is a compromise of the above two schemes. On one hand this broadcasting scheme is relatively simple and no backtracking is required contrasting to approaches using local information. On the other hand collecting limited global information is much less expansive than the approaches using global information. The challenge is to identify the right type of limited global information based on which cost-effective broadcasting can be derived.

In this paper, we study an optimal broadcasting scheme based on an *extended binomial tree* structure and which can tolerate at least $n-2$ link faults. [2] In the proposed scheme, each node keeps limited global information about the faulty links distribution. The concept of *faulty adjacent subcube*, an m-dimensional subcube that contains at least $m-1$ faulty links, is used to represent the basic unit of information. In the absence of faulty links, no information is required to be kept at each node. Results show that the depth of any broadcasting tree is n for an n-dimensional hypercube with no more than $n-2$ faulty links. To our best knowledge, the proposed

[2] The proposed method can also be applied to faulty hypercubes with more than $n-2$ faulty links. However, the optimality cannot be guaranteed.

scheme is the first limited-global-information-based broadcasting that achieves time optimality in injured hypercube with link faults. We also show that $n-2$ is the maximum number of faulty links that can be tolerated in any optimal broadcasting scheme in an n-dimensional hypercube. We also study the fault-tolerant broadcasting under a modified definition of faulty adjacent subcube, where an m-dimensional faulty adjacent subcube is defined as a cube that contains at least m (instead of $m-1$) faulty links. We show that all the results based on the original faulty adjacent subcube definition are still valid under this modified definition of faulty adjacent subcube. The only exception is that the depth of the broadcasting tree in an n-dimensional hypercube with $n-1$ faulty links is n under most cases and is $n+1$ under few cases. Obviously, one more fault can be covered based on this modified definition and one more time step is required in a broadcasting in the worst case. The idea of using limited global information in fault-tolerant broadcasting has also been applied to hypercubes with node faults [18]. However, a different definition of limited global information is used.

This paper is organized as follows. In Section 2 we define basic notation and preliminaries. In Section 3 we propose a fault-tolerant hypercube broadcasting with limited network information, where the faulty adjacent subcube is used as the basic unit of information. An optimal implementation of the proposed scheme is discussed in Section 4. The broadcasting based on another type of limited network information is discussed in Section 5. It is shown that a broadcasting can be completed optimally in n steps, except a few cases with low probability that require $n+1$ steps. Finally, in Section 6 we present some conclusions.

2. Notation and preliminaries

An n-dimensional hypercube (or n-cube) Q_n contains 2^n nodes. Every node a has a binary address $a_n a_{n-1} \dots a_1$, where a_i is called the i-th bit (also called the i-th dimension) of the address. Every m-subcube Q_m has a unique ternary address $u_n u_{n-1} \dots u_1$, with $u_i \in \{0,1,*\}$, and there are exactly m bits take the value $*$, where $*$ is a don't care symbol. The extended Hamming distance between two subcubes $U = u_n u_{n-1} \dots u_1$ and $W = w_n w_{n-1} \dots w_1$ in cube Q_n is defined as $H(U,W) = \sum_{i=1}^{n} h(u_i,w_i)$, where $h(u_i,w_i)$ is 1 only when one and only one of u_i and w_i is 1; otherwise it is 0. For example, $H(*0*,*11) = 1$ and $H(001,100) = 2$. a^i is a node that is adjacent to node a along the i-th dimension. For example, if $a = 1101$ then $a^2 = 1111$. The extended Hamming distance between two subcubes $U = u_n u_{n-1} \dots u_1$ and $W = w_n w_{n-1} \dots w_1$ in cube Q_n is defined as $H(U,W) = \sum_{i=1}^{n} h(u_i,w_i)$, where $h(u_i,w_i)$ is defined in Table 1(a). The *exclusive-or* $\oplus$ operation between two subcubes U and W is defined as $U \oplus W = v_n v_{n-1} \dots v_1$, where $v_i = u_i \oplus w_i$ as shown in Table 1(b).

Definition 1. A Q_n with no fault is called healthy hypercube. A Q_n with at most $n-2$ link faults is called injured hypercube. A Q_n with more than $n-2$ link faults is called faulty hypercube.

Broadcasting is a process which sends a data set from one node (called the source node s) to all the other nodes. The broadcast data visits each node exactly once and forms a *spanning broadcast tree* in the cube. A commonly used spanning tree in hypercubes is the *spanning binomial tree* [2]. A 0-level

Table 1
Definition of (a) $h(u_i,w_i)$, and (b) $u_i \oplus w_i$

(a)				(b)			
$h(u_i,w_i)$ u_i / w_i	0	1	*	$u_i \oplus w_i$ u_i / w_i	0	1	*
0	0	1	0	0	0	1	*
1	1	0	0	1	1	0	*
*	0	0	0	*	*	*	*

binomial tree (B_0) has one node. An n-level binomial trees (B_n) is constructed out of two ($n-1$) level binomial trees by adding one edge between the roots of the two trees and by making either root the new root. We assume that hypercubes under consideration are capable of n-port broadcasting, that is, each node in a cube can communicate with all its neighboring nodes concurrently.

Another view of a binomial tree is proposed in [18], where a Q_n with the source node s is partitioned into $\{Q'_{n-1}, Q'_{n-2}, \ldots, Q'_1, Q'_0, s\}$, such that $d(s, Q'_{n-i}) = 1$, $1 \leq i \leq n$. The sequence $\{c_1, c_2, \ldots, c_n\}$, a permutation of bit positions in Q_n which take value $*$, is called the *coordinate sequence* (CS). This sequence determines the structure of the binomial tree at first level: Q'_{n-i} is connected to s along the c_i-th dimension. Fig. 1 shows such a partition.

The above partition process is also called a *splitting process*. When this process is recursively applied to each element in the partition, it is called a *recursive splitting process*. To be more specific, given a Q_m, a subcube of Q_n, with the source node s and CS $= \{c_1, c_2, \ldots, c_m\}$, the partition can be derived by applying the following *recursive splitting process*: Q'_{m-1} is derived by splitting the Q_m along the c_1-th dimension. The other part Q_{m-1} that contains s will be further split along the c_2-th dimension. Q'_{m-2} is the part which doesn't contain s. This process continues until Q'_1 is split into two nodes, where one is Q'_0 and the other is $Q_0 = s$. Then this process is recursively applied to each cube in $\{Q'_{m-1}, Q'_{m-2}, \ldots, Q'_1\}$ at a node (called *forward node*) which is adjacent to s. We consider the source node s as a special forward node. By connecting source nodes at two subsequent splits, a binomial tree is derived. Note that CS used at each split could be *global* or *local*. If CS used at different nodes are subsequence of one coordinate sequence of n dimensions, then it is global; otherwise, it is local.

More formally, we have the following definition of the splitting process:

Definition 2. A splitting process of Q_m at $s = s_n s_{n-1} \ldots s_1$, with CS $= \{c_1, c_2, \ldots, c_m\}$, forms a partition of Q_m by replacing bit positions in Q_m which take value $*$ by binary values:

$$\begin{aligned} Q_m &= Q_{m-1} + Q'_{m-1} \\ Q_{m-1} &= Q_{m-2} + Q'_{m-2} \\ &\vdots \\ Q_1 &= Q_0 + Q'_0, \end{aligned}$$

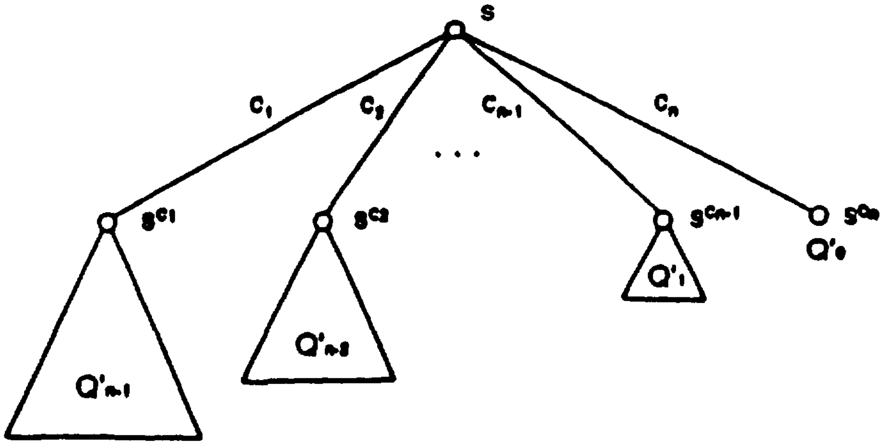

Fig. 1. A partition of Q_n at s with respect to CS $= c_1 c_2 \ldots c_n$.

where Q_{m-i} (Q'_{m-i}) stands for a $(m-i-1)$-cube obtained by replacing the c_i-th bit in Q_{m-i+1} with $s_{c_i}(\bar{s}_{c_i})$, $1 \le i \le m$.

Obviously, $\{Q'_{m-1}, Q'_{m-2}, \ldots, Q'_0, Q_0 = s\}$ is a partition of Q_m.

Proposition 1. *Every two cubes in $\{Q'_{n-1}, Q'_{n-2}, \ldots, Q'_0, s\}$ are adjacent.*

The following definition describes an extended binomial tree structure used in the proposed broadcasting scheme.

Definition 3. Suppose $\{Q'_{n-1}, Q'_{n-2}, \ldots, Q'_0, s\}$ is a partition of Q_n at s following the splitting process defined in Definition 2 and EB_{i-1} is an extended binomial tree of $Q'_i, 1 \le i < n$, and $\mathrm{EB}_0 = Q'_0$. The extended binomial tree EB_n of Q_n with source node s is constructed by adding (arbitrary) $n-1$ edges that connect $\mathrm{EB}_{n-1}, \mathrm{EB}_{n-2}, \ldots, \mathrm{EB}_0, s$ into a connected graph.

Obviously the conventional binomial tree is a special extended binomial tree where all the $n-1$ edges are placed at s. In a partition $\{Q'_{n-1}, Q'_{n-2}, \ldots, Q'_0, s\}$, if the link that connects Q'_{n-i}, $1 \le i \le n$, to s is faulty then Q'_{n-i} is called a *disconnected cube* with respect to s; otherwise, it is a *connected cube*. Obviously, if there is at least one disconnected cube generated from a splitting process in the recursive splitting process then the extended binomial tree is not a conventional binomial tree.

3. Fault-tolerant broadcasting with limited global information

There may not exist a binomial tree B_n with s as the root node in an injured hypercube, since any faulty link that is adjacent to s destroys the corresponding branch originated from s. Then does there exist an extended spanning binomial tree in a given injured hypercube? The answer to this question is shown in the following two theorems. Theorem 1 shows how to find those links to connect elements in any given partition of an injured hypercube. Moreover, the selected links should be close to s so that s can determine the extended binomial tree by using only limited global information, i.e., the distribution of faulty links which are close to s. The selection of those links is shown in the proof of Theorem 1. Theorem 2 ensures that the approach used can be applied recursively to each subsequent partition.

Theorem 1. *There exist $n-1$ healthy links that connect cubes in any partition $\{Q'_{n-1}, Q'_{n-2}, \ldots, Q'_0, s\}$ of an injured cube Q_n. Moreover, these links are either adjacent to or one extended Hamming distance away from s.*

Proof. Suppose there are t healthy links adjacent to s and therefore there are $n-t$ faulty links which connect s to the remaining $n-t$ subcubes in an arbitrarily selected partition. Let Q'_{n-j} be a disconnected cube and Q'_{n-i} be one of the t connected cubes in a partition, we have the following connections among s, Q'_{n-j}, and Q'_{n-i} as shown in Fig. 2(a) and Fig. 2(b). Apparently, there are t node-adjacent detour paths as shown Fig. 2(a): $s \rightarrow s^{c_i} \rightarrow (s^{c_i})^{c_j} \rightarrow s^{c_j}$ when $j < i$, and Fig. 2(b): $s \rightarrow s^{c_i} \rightarrow (s^{c_i})^{c_j}$ when $j > i$ which connect s to a node in Q'_{n-j}. Since there are at most $(n-1)-(n-t) = t-1$ faulty links which are not adjacent to s, there is at least one healthy path among these t paths. Moreover the link connecting Q'_{n-i} and Q'_{n-j} is one extended Hamming distance away from s. □

The next theorem indicates the existence of such a partition in which each subcube is a nonfaulty cube. Therefore, Theorem 1 can be recursively applied at each split, generating an extended binomial tree.

Theorem 2. *There exists a partition* $\{Q'_{n-1}, Q'_{n-2}, \ldots, Q'_0, s\}$ *of an injured hypercube* Q_n *such that* Q'_{n-i}, $0 \leq i \leq n$, *is a nonfaulty subcube, i.e., a healthy or injured subcube. Such a partition is called a safe partition.*

Proof. By induction on n. When $n = 2$ and there exists at most one fault, it is obvious that Q_2 can be partitioned into $\{Q'_1, Q'_0, s, \}$ where Q'_1 is a healthy cube. Assume it is true for all n, such that $n < k$. When $n = k$, determine a dimension i along which there exists at least one faulty link and then split the Q_k into two Q_{k-1}'s along dimension i. Clearly both Q_{k-1}'s contain at most $k - 2$ faults. By using the induction assumption at step 2, we prove Theorem 2 when $n = k$. □

Based on the above Theorems 1 and 2, we have the following result:

Corollary. *There exists an extended binomial tree originated from any node in an injured hypercube.*

With the results of Theorems 1 and 2, we can easily generate a broadcasting algorithm (Algorithm 1). Note that if a subcube Q'_{n-j} is disconnected from the source node s then another cube Q'_{n-i} has to be used to direct the corresponding destination set, which represents the node set in Q'_{n-j}, from s to a node in Q'_{n-j}. Therefore, a forward node (a node in Q'_{n-i}) would receives more than one destination subcube. In general, a forward node a receives $\{Q_m, (Q_{m_1}, b_1), (Q_{m_2}, b_2), \ldots, (Q_{m_k}, b_k)\}$, where Q_m is the subcube to which a belongs. Q_{m_i}, $1 \leq i \leq k$ are disconnected nodes with respect to the parent node of a. b_i, consisting of either one dimension (Fig. 2(b)) or two dimensions (Fig. 2(a)), is a dimension sequence used to direct a broadcast data to each subcube.

Algorithm 1: {fault-tolerant hypercube broadcasting}

{At forward node a with destination cubes $\{Q_m, (Q_{m_1}, b_1), (Q_{m_2}, b_2), \ldots, (Q_{m_k}, b_k)\}$. Initially only the source node s has destination cube Q_n.}

Broadcast Q_m:

1. Find a CS $= \{c_1, c_2, \ldots, c_m\}$ such that the corresponding partition $\{Q'_{m-1}, Q'_{m-2}, \ldots, Q'_0, a\}$ is safe.
2. Send Q'_{m-i} to nodes a^{c_i} if the links between a and a^{c_i} are healthy. For each Q'_{m-j} that the link between s and s^{c_j} is faulty, find a dimension c_i such that one of following two conditions is satisfied:
   (a) $j < i$ and the path $s \rightarrow a^{c_i} \rightarrow (a^{c_i})^{c_j} \rightarrow a^{c_j}$ is healthy.
   (b) $j > i$ and the path $s \rightarrow a^{c_i} \rightarrow (a^{c_i})^{c_j}$ is healthy.
   If condition (a) is true, $(Q'_{m-j}, \{c_j, c_i\})$ is sent to node a^{c_i}. If condition (b) is true, $(Q'_{m-j}, \{c_j\})$ is sent to node a^{c_i}.

Broadcast (Q_{m_i}, b_i), $1 \leq i \leq k$:

1. If $b_i = \{c_j\}$ then send Q_{m_i} to node a^{c_j}, a neighbor along dimension c_j. If $b_i = \{c_j, c_i\}$ then send (Q_{m_i}, c_i) to a^{c_j}.

The proposed scheme performs as a normal binomial-tree-based broadcasting when the hypercube is healthy (and for injured or faulty hypercubes under certain faulty link distributions). In this case, each forward node a only receives one destination cube Q_m, such that $a \in Q_m$. Therefore step 1 of broadcast Q_m of Algorithm 1 is a normal splitting process by randomly selecting a CS. In step 2, each Q'_{m-i} will be directly sent to node s^{c_i}.

Note that if the largest subcube (the left most subcube in Fig. 1) in a partition is a disconnected cube, then one extra step is required. In the worst cast when this situation occurs in the subsequent splitting, a total of $(n - 1)$ extra steps are required to complete a broadcast in Q_n.

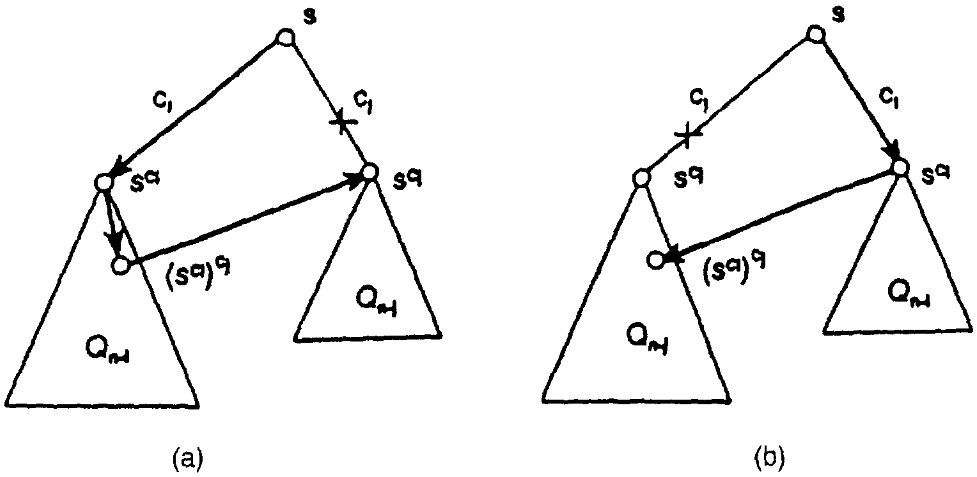

Fig. 2. The connecting path among s, Q'_{n-j} and Q'_{n-i}.

Question. How can each individual partition be performed such that no extra step is required in a broadcasting process?

The answer to the above question is that the link connecting the largest subcube at each partition should be healthy.

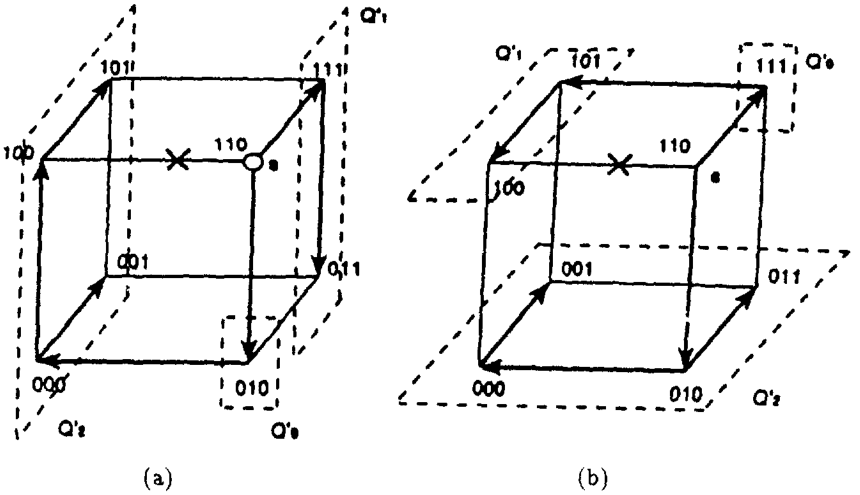

Fig. 3. An injured Q_3 with one faulty link.

Definition 4. A safe partition $\{Q'_{n-1},Q'_{n-2},\ldots,Q'_0,Q_0\}$ of an injured Q_n is optimal if the source node s connects Q'_{n-1} through a healthy link.

Note that when there is no faulty adjacent link, any safe partition is considered optimal. Before the discussion of the implementation of a safe partition, let's first look at the following theorem which shows the depth, i.e., the time steps, of any extended binomial tree generated from an optimal partition at each node of the cube.

Theorem 3. *There exists an extended binomial tree of depth n in an recursive partition of an injured Q_n in which each partition is optimal.*

Proof. We prove the theorem by induction on the dimension of the hypercube. Clearly the theorem holds for any Q_1 and Q_2. Assume that the theorem holds for all Q_n's, with $n<k$. Then for Q_n, with $n=k$, we perform an optimal partition $\{Q'_{k-1},Q'_{k-2},\ldots,Q'_0,Q_0\}$ of Q_k at s. Based on the assumption, broadcasting at each Q'_{k-j}, $1\leq j\leq k$ can be completed in $k-j$ if it is initiated from a node in Q'_{k-j}. Since a disconnected cube Q'_{k-j} can be connected through a path of length two or three, any disconnect cube Q'_{k-j}, $j>2$ requires no more than n steps starting from node s. For a disconnected Q'_{k-2}, we can always find a connected Q'_{k-i}, $2<i\leq k$ to connect Q'_{k-2} to s (since there are at most $k-2$ faulty links in Q_k). Based on Fig. 2 the length of the path connecting Q'_{k-2} to s via Q'_{k-i} is two. Therefore a total of n steps are required based on the induction assumption. □

Consider the Q_3 in Fig. 3, where link $1*0$ is faulty. Suppose a message to be broadcast is generated at node 110. A safe partition of $***$ with respect to CS $=\{2,1,3\}$ is $\{Q'_2=*0*,Q'_1=*11,Q'_0=010,s=110\}$. However, $Q'_2=*0*$ is a disconnected cube and it can be reached via Q'_1 or Q'_0. Fig. 3(a) shows a possible optimal broadcasting where Q'_2 is reached via Q'_0. The number of time steps required is four which is non-optimal. An optimal safe partition of $***$ is with respect to CS $=\{3,1,2\}$ and a possible broadcasting is shown in Fig. 3(b), where Q'_1 is reached via Q'_0. Clearly the number of time steps required is three.

Theorem 3 shows the existence of an extended binomial tree of depth n (optimal) in an injured Q_n with no more than $n-2$ faulty links. Then, does there exist an extended binomial tree of depth n in an injured Q_n with more than $n-2$ faulty links? More strongly, can we prove that $n-2$ is the maximum number in order to maintain optimality for an extended binomial tree, or even stronger for any spanning broadcasting tree? The answer is given in the following theorem.

Theorem 4. *$n-2$ is the maximum number of faulty links that can be tolerated in any optimal broadcasting in an n-cube.*

Proof. It suffices that we provide a counter example. Let's consider a node a in Q_n which is $n-1$ distance away from the source s. If all the $n-1$ faults are adjacent to s and all those faults are along $n-1$ Hamming distance paths from s to a, then a can only be reached through a detour path with one detour, and this path has a length of $(n-1)+2=n+1$. □

The results in this section show mainly the existence of certain features, such as safe partitions in an injured hypercube. A more challenging issue is how to realize these features in an efficient way. More specifically, we shall find an efficient way to perform a safe partition, and, more importantly, an optimal safe partition in a given injured hypercube. These issues are discussed in the next section.

The optimal safe partition describes only the connection of the largest subcube in a safe partition. The distribution of faulty adjacent links that connect other subcubes in the partition is not important if the number of time steps used in a broadcasting is the only concern. Sometimes, it is also desirable that the broadcast data reach each node through a Hamming distance path. In this case, the percentage of nodes in an injured hypercube that receive the broadcast data through a Hamming distance path from the source node is used to measure the quality of a broadcast. This percentage can be controlled by the splitting process at each node. In particular, the placement of adjacent faulty links at each split. The objective here is to reduce the number of detour nodes – nodes that receive the broadcast data through a non-Hamming distance path. Suppose node s in a Q_n has k adjacent faulty links. The placement of these k adjacent faulty links in the CS $= \{c_1, c_2, \ldots, c_n\}$ could be arranged in the following three approaches:

- *Random selection*: in which k adjacent faulty links are randomly placed in k dimensions in CS.
- *Right-first selection*: in which faulty links are placed in k right-most dimensions in CS, i.e., dimensions: $c_{n-(k-1)}, c_{n-(k-2)}, \ldots, c_{n-(k-k)}$.
- *Left-first selection*: in which faulty links are placed in the k left-most dimensions in CS, i.e., dimensions $c_1, c_2, \ldots, c_k$.

It has been shown [15] that the right-first selection outperforms both the left-first and the random selections in reducing the number of detour nodes. Therefore, an optimal safe partition at a node a should be determined such that faulty adjacent links of a are used to connect small subcubes in the partition. However, the strict right-first selection may not always be possible for certain configurations of injured hypercubes. For example, in an injured Q_n with four faulty links located in a Q_2 that contains the source node s, there are two possible placements of two adjacent faulty links to two right-most dimensions in a partition, but neither of them is safe. A relaxed version of the right-first selection can be adopted where adjacent faulty links are used to connect small subcubes in the partition as long as they do not violate the safe partition condition. This approach is implicitly used in the implementation of the proposed scheme as described in the next section.

4. Implementations

In this section, we describe an efficient implementation of the proposed broadcast algorithm, and examine the type and the amount of limited global information required at each node.

Obviously, the way to perform a safe partition is the key. One possible method is based on the proof of Theorem 2, where at each split of the cube at least one faulty link must be along the splitting dimension, i.e., the remaining subcube is nonfaulty. There are two flaws with this approach. First, each node needs to know global information to locate faulty links in the cube. In general, it is expansive to maintain an updated global information on each node. Second, optimality can not be guaranteed. For example, suppose the source s has one adjacent faulty link (along dimension d_1) and it is the only faulty link in the injured Q_n, then the dimension d_1 has to be selected as the first element of CS. In this case Q'_{n-1} is a disconnected cube. We try to determine an optimal safe partition method which uses limited global information. Moreover, this method should be efficient, which requires a time complexity in an order of $O(m)$ to determine a CS of length m at each node.

At each node, locations of adjacent faulty subcubes and dimensions along which faulty links are located in these subcubes are kept as limited global information associated with each node. Adjacent faulty links are considered to be an adjacent faulty Q_0. The process that collects information of adjacent faulty subcubes at each node is beyond the scope of this paper. This process can be determined following a similar approach used in [19]. Note that such an information collection process is not necessary in the absence of faulty links, i.e., this process is activated only when one or more faulty links are detected.

Let $I = (Q_{m_1},d_1),(Q_{m_2},d_2),\ldots,(Q_{m_k},d_k)$ be the *faulty adjacent subcube list* attached to node a, that is $H(Q_{m_i},a) = 1$. Q_{m_i} in the tuple (Q_{m_i},d_i) represents the absolute address of a subcube, and the number of $*$ in each subcube determines the size of the subcube with the exception of one $*$ which could represent a faulty adjacent Q_0 or Q_1. For example, $Q_{m_i} = **1$ is an adjacent 2-cube to $s = 010$. The location of 1 in $s \oplus Q_{m_i} = **1$ represents the dimension along which Q_{m_i} and s are connected. If there is no occurrence of 1 in $s \oplus Q_{m_i}$ then Q_{m_i} is a faulty adjacent link. d_i, *faulty dimension set*, is a set of dimensions along which faults in Q_{m_i} occur. In Fig. 3, the d_i for $1*0$ is $\{2\}$. Apparently, when Q_{m_i} is a 0-cube or a 1-cube, d_i is not necessary.

The basic idea used in the implementation of step 1 of broadcast Q_m in Algorithm 1 is to ensure that all the subcubes in the partition of Q_m are nonfaulty (injured or healthy) and Q'_{m-1} is a connected subcube to achieve an optimal safe partition. The challenge here is to determine an optimal partition of a destination subcube Q_m in an order of $O(m)$.

The following notation is used in the implementation:

A: A set of dimensions of adjacent faulty links. Those dimensions are not contained in faulty dimension sets in other faulty adjacent cubes.

AF: A set of dimensions of adjacent faulty links and those dimensions are contained in faulty dimension sets in other faulty adjacent cubes.

F: A union of faulty dimension sets excluding those dimensions in AF.

N: A set of dimensions not in A, AF or F.

Clearly, A, AF, F, and N can be directly derived from the union of the faulty dimension sets ($\bigcup d_i$) in I and faulty adjacent links. Let a, af, f, n denote elements in A, AF, F, N, respectively. $|X|$ represents the cardinality of the set X. $|X|_y$ is a random sequence of y with a length $|X|$. $(X)_y \cdot (X')_{y'}$ is a concatenation of two dimension sequences. The optimal selection of $\mathrm{CS} = c_1c_2\ldots c_m$ at each node to partition a subcube Q_m can be described using the following scheme:

$$c_1c_2\ldots c_m = |F|_f \cdot \alpha_n \cdot |AF|_{af} \cdot \beta_n \cdot |A|_a$$

where

$$\alpha = \min\{|N|,|AF| + |A|\}$$

and

$$\beta = \begin{cases} 0 & \text{if} \quad \alpha = |N|, \\ |N| - \alpha & \text{if} \quad \alpha = |AF| + |F|. \end{cases}$$

In the above equation, the A, AF, F, and N in the above equation are subsets of respective A, AF, F, N which contain only those dimensions with $*$ in the destination cube. Clearly, with the randomness used in the selection process, the above scheme determines a CS that corresponds to an optimal partition in $O(m)$.

Theorem 5. *The partition of an nonfaulty Q_n using the above scheme is safe and optimal.*

Proof. This theorem can be proved in two steps. We first prove that each Q'_i, $1 \le i \le n-1$, in $Q_n = \{Q'_{n-1},Q'_{n-2},\ldots,Q'_1,Q'_0,Q_0\}$ is a nonfaulty cube and Q'_{n-1} is a connected cube. Therefore the partition is safe and optimal.

To prove that all subcubes in any partition are nonfaulty (which are either injured or healthy), we need to use the following two facts: (1) If a Q_n is an injured hypercube and it is split along a dimension along which there exists a faulty link, then both Q_{n-1} and Q'_{n-1} in $Q_n = Q_{n-1} + Q'_{n-1}$ are nonfaulty. (2) Any faulty adjacent link of node s is not contained in subcubes in any partition with node s as the source node. Based on fact (1), we can first select those dimensions in F and the resulting subcubes are guaranteed nonfaulty. Using fact (2), we can then select α ($\le |AF| + |A|$) dimensions from N without generating faulty subcubes. Dimensions from AF are selected without generating faulty subcubes because each dimension in AF contains at least two faulty links, although the rest of the selection from A and the remaining N can be random. We give priority to dimension from N to reduce the number of detour nodes.

The partition is optimal since $|F| + |AF| + |A| \geq 1$ in an injured hypercube, i.e., no dimension from A or AF is selected as the first element in CS. □

Since the union of faulty dimension sets is available at each node, together with the information of faulty adjacent links, we can easily derive A, AF, F, and N. The CS can then be determined through a simple concatenation of random sequences of dimensions from A, AF, F, and N.

An optimal partition does not automatically guarantee an optimal broadcasting. Care should be token to connect each disconnected subcube via a connected subcube. More specifically, in a partition of Q_n, we should connect the disconnected Q'_{n-2} via a connected subcube with a smaller dimension, i.e., Q'_{n-2} is connected via a path of length 2 form the source. This selection is reflected in Algorithm 2, an implementation of step 2 of broadcast Q_m in Algorithm 1, where the distribution of faulty adjacent 0-cubes (or faulty adjacent links) and faulty adjacent 1-cubes are the only information required.

Algorithm 2: {Implementation of Algorithm 1, step (2) of broadcast Q_m}

{Suppose an optimal safe partition $\{Q'_{m-1}, Q'_{m-2}, \ldots, Q'_1, a\}$ is given at node a}

1. For each Q'_{m-i} with healthy adjacent 0-cube along c_i, send Q'_{m-i} to node a^{c_i} along c_i.
2. For each remaining Q'_{m-j} with each faulty adjacent 0-cube along c_j, find a nonfaulty 2-cube containing node a that spans along dimensions c_j and c_i. When $j = 2$, i should be selected such that $2 < i < n$. Such a Q_2 can be determined by examining the adjacent 0-cubes and 1-cubes of node a.
3. Send Q'_{m-j} together with c_j, (or c_j, c_i) to node a^{c_j} when $j > i$ (when $i > j$).

Table 2 shows the faulty dimension set associated with each node of a Q_4 in Fig. 4, with two faulty links $1*01$ and $100*$. Six faulty subcubes are $100*$, $1*01$, $1*01$, $1**1$, $10**$, $1***$. The A, AF, F, and N for each node is also shown together with the CS that generates optimal safe partition at each node. Suppose node 0001 is the source node, the CS: $\{1,3\} \cdot \{2,4\}$ represents four possible selections: 1324, 1342, 3124, and 3142. For example, CS = 1324 generates the optimal safe partition $\{***0, *1*1, *011, 1001, 0001\}$. Actually, each subcube in the partition is a healthy one. Therefore the optimality is guaranteed which is independent of the subsequent splitting processes. Fig. 4 shows a possible broadcasting in this cube, where the subsequent splitting processes on each subcube are based

Table 2
Faulty adjacent hypercube set

Nodes	$\bigcup d_i$	A	AF	F	N	Optimal CS
$0***$	{1,3}	{ }	{ }	{1,3}	{2,4}	$\{1,3\} \cdot \{2,4\}$
1000	{1,3}	{1}	{ }	{3}	{2,4}	$\{3\} \cdot \{2,4\} \cdot \{1\}$
1001	{1,3}	{1,3}	{ }	{ }	{2,4}	$\{2,4\} \cdot \{1,3\} \cdot$
$1*1*$	{1,3}	{ }	{ }	{1,3}	{2,4}	$\{1,3\} \cdot \{2,4\}$
1100	{1,3}	{ }	{ }	{1,3}	{2,4}	$\{1,3\} \cdot \{2,4\}$
1101	{1,3}	{3}	{ }	{1}	{2,4}	$\{1\} \cdot \{2,4\} \cdot \{3\}$

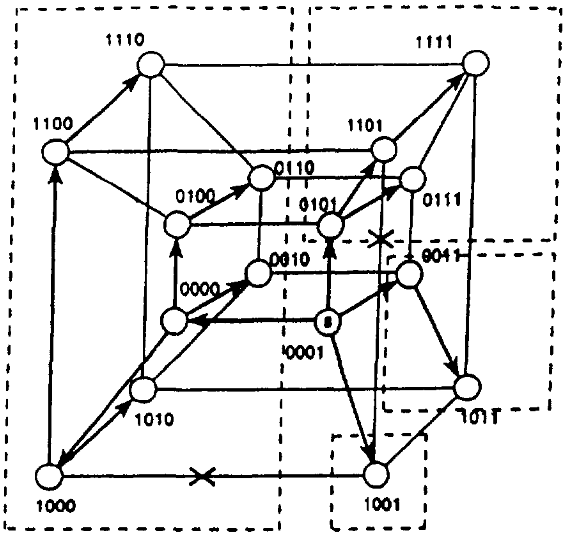

Fig. 4. An injured hypercube Q_4 with two faulty links.

on the global CS = 4321. For other CSs at node 0001, they may or may not generate a broadcasting tree with a depth equals to the dimension of hypercube. For example, it is easy to see that any CSs in the format of $\{4\} \cdot \{1,2,3\}$ will generate a broadcasting tree of depth at least five. On the other hand, the CS = 1423 at 0001 could generate a broadcast tree of depth four, although the corresponding partition is not safe (the Q'_2 is a faulty hypercube).

The amount of limited algorithm information can be further reduced by maintaining information of faulty adjacent subcubes whose dimensions are smaller than two (based on Algorithm 2, the implementation of step 1 of broadcast Q_m). For other faulty adjacent subcubes, only the union of their faulty dimension sets $(\bigcup d_i)$ in the faulty adjacent subcube list is required to determine the optimal safe partition.

5. Extensions

In this section, we study another way of defining limited global information. Actually, we still use the same concept of the faulty adjacent subcube but it is defined differently. The main objective is to get insights into different trade-offs. In this case, the performance (in terms of the number of time steps in a broadcasting) is traded by the degree of fault tolerance. To be more specific, the scheme studied in this section can tolerate $(n-1)$ faulty links (one more fault to be tolerated than the previous scheme) and a broadcasting in an injured Q_n still can be completed optimally in n steps, except in a few cases with low probability which require $n+1$ steps.

The following is the modified Definition 1:

Definition 1′. A Q_n with no fault is called healthy hypercube. A Q_n with at most $n-1$ link faults is called injured hypercube. A Q_n with more than $n-1$ link faults is called faulty hypercube.

It is easy to verify that all theorems and algorithms in the previous sections are still applicable to injured hypercubes under the above definition. The proof is left to the reader. The only exception is that Theorem 3 does not hold any more. It can be rewritten as the following theorem:

Theorem 3′. *Any extended binomial tree generated from an optimal partition at each node of an injured hypercube has a depth of n or n + 1.*

Proof. We prove the theorem by induction on the dimension of the hypercube. Clearly the Theorem holds for any Q_1 and Q_2. Assume that this theorem holds for all Q_n's, with $n < k$, then for Q_n, with $n = k$, we perform an optimal partition $\{Q'_{k-1}, Q'_{k-2}, \ldots, Q'_0, Q_0\}$ of Q_k at s. Based on the assumption, each Q'_{k-i}, $1 \le i \le k$, can be completed in $k-i$ and $k-i+1$ steps. Since cube Q'_{k-1} is connected through a healthy link, $(k-1)+1$ or $(k-1)+1+1 = k+1$ steps are required to complete broadcasting among nodes in Q'_{k-1}. Since the other Q'_{k-i} may be connected through faulty links, the worst case, in terms of the longest path from s to a leave, occurs when Q'_{n-2} (a healthy subcube) or Q'_{n-3} (an injured subcube) is connected to s by a faulty link. In the former case, Q'_{n-2} can be reached only through a detour path. Since Q'_{n-2} is a healthy subcube, $(n-2)$ additional steps are required to complete broadcasting inside Q'_{n-2}. Therefore, a total of $(n-2)+1+2=n+1$ steps are required. In the later case, Q'_{n-3} can be reached only through a detour path. Since Q'_{n-3} is an injured hypercube and based on the induction assumption, a total of $((n-3)+1)+1+2=n+1$ steps are required in the worst case. □

Clearly both Algorithms 1 and 2 can be used here. To estimate the probability of generating a broadcasting tree of depth $n+1$, we have the following observations: Suppose s is the source node in a destination subcube Q_m, an extra step is required when all the $m-1$ faults in Q_m are adjacent to s. This extra step will cause an extra global time step only when the source node (in a subsequent split) is a node in the longest path in the broadcasting tree (the left-most path in the tree).

A simulation study has been conducted on Q_n's with $n-1$ faulty links, where n ranges from 2 to 10. Under the assumption of random distribution of faulty links, we obtain the percentage of non-optimal broadcasting for cubes of different sizes as follows: 50% for Q_2, 4% for Q_3, 0.1% for Q_4, 0.01% for Q_5, 0.0001% for Q_6, and close to 0% for Q_7, Q_8, Q_9, and Q_{10}. Clearly, the percentage of non-optimal broadcasting decreases drastically as the size of the cube increases.

6. Conclusions

In this paper, we have studied an optimal broadcasting scheme that can tolerate $n-2$ faulty links. This process is based on an extended spanning binomial tree structure which keeps the simplicity of conventional binomial-tree-based broadcasting. We have also shown that $n-2$ is the maximum number of faulty links that can be tolerated to achieve an optimal broadcasting. The proposed broadcasting can be achieved efficiently and each node only needs to know limited global information captured in a faulty adjacent subcube list.

There are other issues that have not been included but can be treated separately. One issue is the deadlock problem which could happen when multiple nodes send their broadcast data simultaneously. In general, deadlock could be avoided using message buffers and virtual channels [3] at a lower level of implementation. The integration of the deadlock-freeness feature and different communication schemes has been discussed in [7].

Acknowledgements

Thanks to Tianan Wu for providing help in obtaining simulation results.

References

[1] A. Al-Dhelaan and B. Bose, Efficient fault tolerant broadcasting algorithm for the hypercube, *Proc. of 4th Conf. on Hypercube Concurrent Computers and Applications* (1989) 123–128.

[2] M.R. Brown, Implementation and analysis of binomial queue algorithms, *SIAM Journal of Computing* (Aug. 1978) 161–164.

[3] W.J. Dally and C.L. Seitz, Deadlock-free message routing in multiprocessor interconnection networks, *IEEE Transactions on Computers* 36(5) (1987) 547–553.

[4] J.P. Hayes and T.N. Mudge, Hypercube supercomputers, *Proceedings of the IEEE* 77(12) (1989) 1829–1841.

[5] W.D. Hills, *The Connection Machine* (MIT Press, Cambridge, MA, 1985).

[6] S.L. Johnson and C.-T Ho, Optimal broadcasting and personalized communication in hypercubes, *IEEE Transactions on Computers* 41(10) (1992) 1249–1268.

[7] T.C. Lee and J.P. Hayes, A fault-tolerant communication scheme for hypercube computers, *IEEE Transactions on Computers* 41(10) (1992) 1242–1256.

[8] Z. Li and J. Wu, A multidestination routing scheme for hypercube multiprocessors, *Proceedings of 1991 International Conference on Parallel Processing*, Vol. II, Aug. 1991, 290–291.

[9] M. Perrcy and P. Banerjee, Distributed algorithms for shortest-path, deadlock-free routing and broadcasting in arbitrarily faulty hypercubes, *Proceedings of 20th International Symposium on Fault-Tolerant Computing*, 1990, 218–225.

[10] C.S. Raghavendra, P.J. Yang and S.B. Tien, Free dimensions – An effective approach to achieving fault tolerance in hypercubes, *Proceedings of 22nd International Symposium on Fault-Tolerant Computing*, 1992, 170–177.

[11] P. Ramanathan and K.G. Shin, Reliable broadcast in hypercube multicomputers, *IEEE Transactions on Computers* 37(12) (1988) 1654–1657.

[12] Y. Saad and M.H. Schultz, Topological properties of hypercubes, *IEEE Transactions on Computers* 37(7) (1988) 867–872.

[13] C.L. Seitz, The cosmic cube, *Communications of the ACM* 28(1) (1985) 22–33.

[14] H. Sullivan, T. Bashkow and D. Klappholz, A large scale, homogeneous, fully distributed parallel machine, *Proceedings of 4th Annual Symposium on Computer Architecture*, March 1977, 105–124.

[15] J. Wu, Broadcasting in injured hypercubes using incomplete spanning binomial trees, Technical Report, TR-CSE-92-29, Department of Computer Science and Engineering, Florida Atlantic University, Nov. 1992.

[16] J. Wu, Fault-tolerant nonredundant broadcasting in Hypercubes, *Proceedings of the 21th International Conference on Parallel Processing*, Sept. 1992, 23–26.

[17] J. Wu and E.B. Fernandez, Broadcasting in faulty cube-connected-cycles with minimum recovery time, *Proceedings of CONPAR92*, Springer-Verlag, LNCS 634, Sept. 1992, 833–834.

[18] J. Wu and E.B. Fernandez, Reliable broadcasting in faulty hypercube computers, *Microprocessing and Microprogramming*, 39 (1993) 43–53.

[19] J. Wu, Broadcasting in Injured Hypercubes using incomplete spanning binomial trees, *IEEE Trans. Comput.* 44(5) (May 1995) 702–705.

Jie Wu received a B.S. Degree in computer engineering in 1982, an M.S. in computer science in 1985, both from Shanghai University of Science and Technology, Shanghai, People's Republic of China, and a Ph.D. in computer engineering from Florida Atlantic University, Boca Raton, Florida, in 1989. During 1985–1987, he taught at Shanghai University of Science and Technology. Since August 1989, he has been with the Department of Computer Science and Engineering, Florida Atlantic University, where currently he is a tenured Associate Professor. He has authored/co-authored over 60 technical papers in various journals and conference proceedings including *IEEE Transactions in Software Engineering*, *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Computers*, *Journal of Parallel and Distributed Computing*, and *Parallel Processing: Practice and Experience*. His research interests are in the area of fault-tolerant computing, distributed algorithms, interconnection networks, Petri net applications, and computer security. Dr. Wu is member of Upsilon Pi Epsilon and ACM, and a senior member of IEEE. He has been on the program committees of the IEEE International Conference on Distributed Computing Systems and International Conference on Computer Communications and Networks, among others.