Broadcasting in Injured Hypecubes Using Limited Global Information

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

Abstract

We propose a fault-tolerant broadcasting algorithm in hypercubes with link faults. This algorithm is based on an extended spanning binomial tree structure which keeps the simplicity of conventional binomial-tree-based broadcasting. Each node keeps limited information of nearby faulty links in terms of faulty adjacent subcubes. It is shown that under most circumstances, a broadcasting can be completed optimally in n steps except few cases with low probability which require $n + 1$ steps.

1 Introduction

We investigate fault-tolerant broadcasting in hypercubes [7] with faulty links. Broadcasting [3] concerns transmitting a data set from one node to all the other nodes in a network. Broadcasting is a very important operation frequently used in a variety of linear algorithms, database queries, and linear programming algorithms. The standard broadcasting algorithm is based on the binomial tree structure. Fault-tolerant broadcasting deals with successful broadcasting in the presence of faulty components (links and/or nodes), and it can be classified based on the following parameters: (1) The way each destination receives the broadcast data. (2) The amount of information kept at each node. (3) The type of faulty components. (4) The number of faulty components.

Normally, broadcasting algorithms should be designed such that the broadcast data is sent to each node once and only once. In such an algorithm, the amount of fault information kept at each node can be classified as local, limited, and global. Local information contains only adjacent faulty components. Limited 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.

The fault-tolerant broadcasting [1] based on local information normally requires routing history as part of message to be broadcasted in order to reach each node once and only once. The fault-tolerant broadcasting [6], [9] based on global information, although has its merit of simplicity, requires a process which collects global information. The broadcasting based on limited information is a compromise of the above two schemes. On one hand this broadcasting scheme is relatively simple and no backtracking is required. On the other hand collecting limited information is much less expensive than the approaches using global information.

In this paper, we study a broadcasting scheme which is based on an *extended binomial tree* structure and which can tolerate at least $n-1$ link faults. We also show that under most circumstances a broadcast can be completed optimally with n steps, and in the worst case it can be completed in $n+1$ steps. In the proposed scheme, each node keeps limited global information about faulty links distribution. The concept of *faulty adjacent subcube*, an m-dimensional subcube that contains at least m faulty links, is used to represent the basic unit of information. In the absence of faulty links, no information is required at each node. We also evaluate the length of the broadcasting path for each destination node under the worst fault distribution. Similar idea has been applied to hypercubes with node faults [4], [10]. However, a different definition of limited global information is used.

With space limitation, all the proofs of theorems in this paper are omitted. The details of all the proofs can be found in [8].

0-7803-1797-1/94/$3.00 © 1994 IEEE

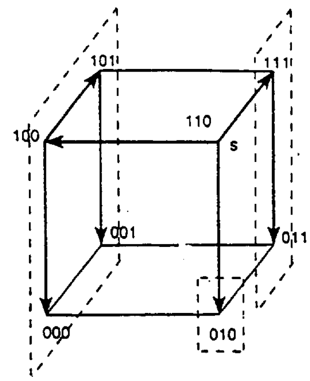

Figure 1: A 3-cube (Q_3)

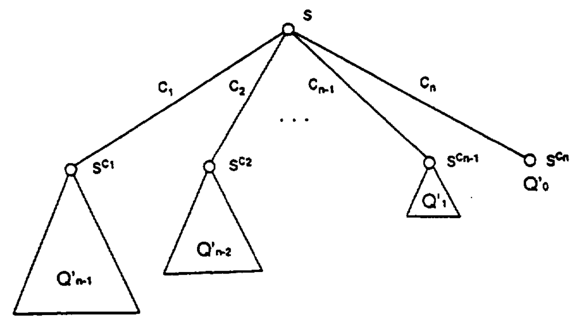

Figure 2: A partition of Q_n at s w.r.t. $cs = c_1c_2...c_n$

2 Notation and Preliminaries

An n-dimensional hypercube (or n-cube) Q_n contains 2^n nodes. Every node a has a binary address $a_na_{n-1}\cdots a_1$, where a_i is called the ith bit (also called the ith dimension) of the address. Every subcube Q_m has a unique trinary address $u_nu_{n-1}\cdots u_1$, with $u_i \in \{0,1,*\}$, and there are exactly m bits take the value $*$, where $*$ is a don't care symbol. Fig.1 shows a 3-cube and three of its subcubes $*0*$, $*11$ and 010. a^i is a node that is adjacent to node a along the ith dimension. For example, if $a = 1101$ then $a^2 = 1111$. 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 common spanning tree used in hypercube broadcast is the *spanning binomial tree* [2]. A 0-level 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. Another view of binomial tree is proposed in [5] 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_ith dimension. Fig. 2 shows such a partition.

The 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 $cs = \{c_1, c_2, ..., c_m\}$, 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_1th dimension. The other part Q_{m-1} which contains s will be further split along the c_2th dimension. Q'_{m-2} is the part which doesn't contain s. This process continues until Q'_1 is split into two nodes, one is Q'_0 and the other is $Q_0 = s$. Then this process is recursively applied to each cube Q'_{m-i} in $\{Q'_{m-1}, Q'_{m-2}, ..., Q'_1\}$ at the node (the new source node) which is adjacent to s in Q'_{m-i}. By connecting source nodes at two subsequent splits, a binomial tree is derived. Fig.1 shows the splitting process of Q_3 at $s = 110$ with $cs = \{2,1,3\}$, and it generates a partition $\{*0*, *11, 010, 110\}$. Similarly the splitting process is applied at $s = 100$ in $*0*$ with $s = \{1,3\}$ and at $s = 111$ in $*11$ with $cs = \{3\}$. This process continues until each subcube becomes a 0-cube. The resultant binomial tree is shown in Fig.1. Clearly, any two cubes in $\{Q'_{n-1}, Q'_{n-2}, \ldots, Q'_0, s\}$ are adjacent.

Definition 1: *Suppose $\{Q'_{n-1}, Q'_{n-2}, \ldots, Q'_0, s\}$ is a partition of Q_n at s following the splitting process and EB_{i-1} is an extended binomial tree of $Q'_i, 1 \leq i < n$ and $EB_0 = Q'_0$. The extended binomial tree EB_n of Q_n with source node s is formed by adding (arbitrary) $n-1$ edges that connect $EB_{n-1}, EB_{n-2}, \ldots, EB_0, s$ to form a connected graph.*

Obviously the conventional binomial tree is 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 $Q'_{n-i}, 1 \leq i \leq n$, 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, then the extended binomial tree cannot be a conventional binomial tree. A *detour* is a dimension which is not in the relative address of two nodes. A node is called *detour node* if it receives the broadcast data from a non-Hamming distance path,

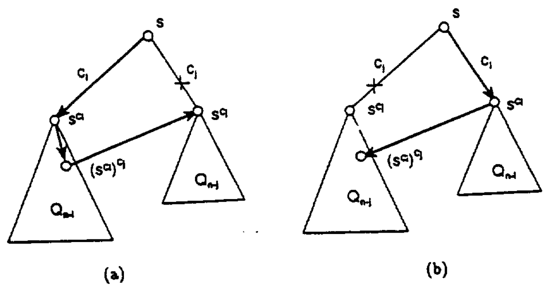

Figure 3: Connection paths among s, Q'_{n-j} and Q'_{n-i}

i.e. detour path, initiated from the source s. In general, a detour path contains several detours each of which generates two extra steps.

3 The Proposed Algorithms

Suppose s is the source node, there may not exist a binomial tree B_n with s as the root node in an injured Q_n, since any faulty link that connects s destroys the corresponding branch originated from s. Theorem 1 shows how to find those links that connect subcubes in any given partition of an injured hypercube. Moreover, those links should be close to s so that s can determine the extended binomial tree by using only limited information. Theorem 2 ensures that the above approach can be applied recursively.

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 hop away from s.*

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

Corollary : *There exists an extended binomial tree for any safe partition $\{Q'_{n-1}, Q'_{n-2}, \ldots, Q'_0, Q_0\}$.*

Note that if a subcube Q'_{n-j} cannot be directly connected to the source node s then another cube Q'_{n-i} has to be used to direct the destination set, which represents the node set in Q'_{n-j}, from s to a node in Q'_{n-j}. Therefore, an intermediate node (a node in Q'_{n-i}) would receive more than one destination subcube. In general, an intermediate 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 is belong. $Q_{m_i}, 1 \le i \le k$, are disconnected nodes with respect to the parent node of a. b_i, a string consisting of either one dimension (Fig. 4 (b)) or two dimensions (Fig. 4 (a)), is a sequence of dimensions along which these subcubes can be reached. Initially only the source node s has one destination cube Q_n.

Algorithm 1: {fault-tolerant hypercube broadcasting}
Broadcast Q_m:

1. Find a cs such that the corresponding partition $\{Q'_{m-1}, Q'_{m-2}, \ldots, Q'_0, a\}$ is safe.
2. Send Q'_{m-i} to node a^{c_i} for all those a^{c_i} such that the link between a and a^{c_i} is healthy. Let Q'_{m-j} be those cubes that the link between s and s^{c_j} is faulty. For each Q'_{m-j} find an i such that one of following two conditions is satisfied: (a) $i < j$ and the path $s \longrightarrow a^{c_i} \longrightarrow (a^{c_i})^{c_j} \longrightarrow a^{c_j}$ is healthy. (b) $i > j$ and the path $s \longrightarrow a^{c_i} \longrightarrow (a^{c_i})^{c_j}$ is healthy. If condition (a) is true, $(Q'_{m-j}, \{c_j, c_i\})$ will be sent to node a^{c_i}. If condition (b) is true, $(Q'_{m-j}, \{c_j\})$ will be sent to node a^{c_i}.

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

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. In this case, each intermediate node a will only receive one destination cube Q'_m, such that $a \in Q_m$. Therefore step 1 of broadcast Q_m is a normal splitting process with a random selection of cs. In step 2, each Q'_{m-i} will be directly sent to node s^{c_i}.

4 Implementations

To determine the minimum amount of information required at each node to implement Algorithm 1, we keep at each node the locations of adjacent faulty subcubes and dimensions in which faulty links are located in these subcubes. Adjacent faulty links are considered as an adjacent faulty Q_0. The process that collects information of adjacent faulty subcubes at each node is beyond the scope of this paper. Note that such a 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 $H(Q_{m_i}, a) = 1, 1 \le i$. Q_{m_i} in the tuple (Q_{m_i}, d_i) represents the absolute address of a subcube. d_i, *faulty dimensions*, is a set of dimensions along which faults in Q_{m_i} occur. In Fig.4, the d_i for

$*0*$ is $\{3\}$. Apparently, when Q_{m_i} is a 0-cube or 1-cube, d_i is not necessary.

The basic idea used to implement the step 1 of broadcast Q_m is as follows: Based on faulty adjacent subcube set I, the dimensions in cs is determined recursively in the order of $c_1, c_2, \ldots, c_m$ using the splitting process defined in Section 1. First of all, if $I \neq \phi$ then a nonfaulty $(m-1)$-subcube (Q'_{m-1}) is selected which is connected to s along c_1 and there is at least one faulty link along dimension c_1. The later requirement ensures that the remaining Q_{m-1} cube is nonfaulty and therefore this process can proceed on Q_{m-1}. If $I = \phi$ then any partition is safe in the absence of faulty adjacent subcubes. Similarly c_2 in cs can be derived from Q_{m-1}, based on an updated I which includes only faulty adjacent subcubes within Q_{m-1}. Following the above procedure all the $c_i, 1 \leq i \leq m$ can be derived.

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

1. Copy the current I to I'
2. If $I' = \emptyset$ then cs will be randomly selected as in normal binomial tree broadcasting. If $I' \neq \emptyset$ then $cs = \{c_1, c_2, \ldots, c_m\}$ will be recursively defined as follows:
   (a) c_1 is selected such that it is included in I', that is, c_1 is registered as a dimension along which there exist at least one fault. Update I' by deleting all the faulty cubes, together with the corresponding faulty dimensions, which don't belong to $Q_{m-1} = (Q_m)^{c_1}_{s_{c_1}}$ [1].
   (b) The rest of $c_i, 2 \leq i \leq m$, are determined as follows: if $I' = \phi$ then the rest of c_i will be randomly selected; otherwise, c_i are determined in sequence following step (a) by replacing appropriate parameters.

Theorem 3: *The partition generated from Algorithm 2 is safe, and it is applicable to all the injured hypercubes.*

Algorithm 3: {Implementation of Algorithm 1, step (2) of broadcast Q_m }

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

1. For each Q'_{m-i} with a 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 a faulty adjacent 0-cube along c_j, find a nonfaulty 2-cube, containing node a, that spans dimensions c_j and c_i. Such a Q_2 can be determined by examining all the adjacent 0-cubes and 1-cubes at node a. Send Q'_{m-j} together with $\{c_j\}$, (or $\{c_j, c_i\}$) to node a^{c_i} when $i > j$ (when $j > i$).

[1] $(Q_m)^{c_1}_{s_{c_1}}$ denotes a cube generated by replacing the occurrence of c_1 by s_{c_1}.

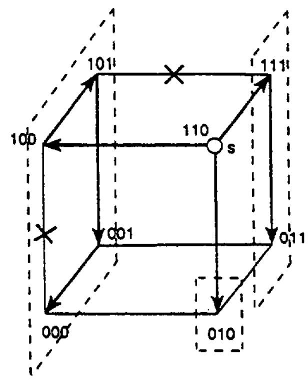

Figure 4: An injured hypercube with two faulty links

Consider the Q_3 in Fig.4, where links $1*1$ and $*00$ are faulty. Suppose a message to be broadcasted is generated at node 110 with a faulty adjacent subcube set $I = \{10*\}$. A safe partition of $***$ with respect to $cs = \{2, 1, 3\}$ is $\{Q'_2 = *0*, Q'_1 = *11, Q'_0 = 010, s = 110\}$. Since all the adjacent links of s are healthy, nodes $100, 111, 010$ receive Q'_2, Q'_1 and Q'_0, respectively. At node 100 which has $I = \{10*\}$ with destination set $Q'_2 = *0*$. The coordinate sequence $cs = \{1, 3\}$ determines a safe partition $\{00*, 101, 100\}$. The resultant broadcasting tree is shown in Fig.4. Since the link between 100 and $00*$ is faulty, the broadcast data reaches $00*$ via node 101.

5 Performance Analysis

In this section, we study the selection of partition scheme to minimize the total number of detour nodes. That is, we try to maximize the percentage of nodes that receive the broadcast data through a Hamming distance path from the source node. The number of detour nodes can be controlled by the splitting process at each node. In particular, the placement of adjacent faulty links at each split. Since there is no relationship between two splits and each node keeps only limited global network information, it is impossible to global minimizing the number of detour nodes. The key point here is the placement of adjacent faulty links at each splitting process to achieve local optimality. 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 $c_1, c_2, \ldots, c_k$.

The recursive splitting process (Algorithm 2) could be used to implement the left-first selection and the random selection (to a certain extend). While the *reverse recursive splitting* [8] could be applied for the right-first selection and the random selection. The following theorems show the relationship between the placement of adjacent faulty links and the number of induced detour nodes.

Theorem 4: *Given a fixed number of adjacent faulty links k at node a, the minimum number of detour nodes generated at node a is $2^k - 1$. This can be achieved by using the right-first selection.*

Theorem 5: *Given a fixed number of faulty adjacent links k at node a, the maximum number of detour nodes that can be generated at a is $2^{n-k-1}(2^k - 1)$, where n is the size of cube. This occurs when the left-first selection is used.*

Based on the above results, it is clear that the right-first selection outperforms the left-first selection in terms of the number of detour nodes generated. The next definition defines a special safe partition that leads to an optimal broadcasting.

Definition 2: *A safe partition of $\{Q'_{n-1}, Q'_{n-2}, \ldots, Q'_0, Q_0\}$ of injured Q_n is optimal if the source node s connects Q'_{n-1} through a healthy link. In addition, the source node s connects Q'_{n-2} through a faulty link only when all the faults in Q_n are adjacent to s.*

Theorem 6: *An extended binomial tree generated from an optimal safe partition at each node of an injured hypercube has a height of n or $n+1$.*

Since the diameter of an injured hypercube is either n or $n+1$, the above result is optimal. The algorithm that determines an optimal safe partition can be found in [8].

6 Conclusions

We have proposed a reliable fault-tolerant hypercube broadcasting algorithm in hypercubes with link faults. This process is based on an extended binomial tree structure which keeps the simplicity of conventional binomial-tree based broadcasting. In addition, it is efficient in the sense that no backtracking is required in the broadcasting process. Each node keeps limited information of nearby faulty links in terms of faulty adjacent subcubes in which the number of faulty links contained is larger than the cube dimension. It is shown that a broadcasting using this scheme requires n steps in an n-cube for the most cases, and in the worst case it requires n+1 steps.

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] S. L. Johnsson and C.-T. Ho. Optimal broadcasting and personalized communication in hypercubes. *IEEE Transactions on Computers.* 38, (9), Sept. 1989, 1249-1268.

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

[5] Z. Li and J. Wu. A multidestination routing scheme for hypercube multiprocessors. *Proc. of 1991 International Conf. on Parallel Processing.* Vol. II, Aug. 1991, 290-291.

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

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

[8] J. Wu. Broadcasting in injured hypercubes using limited global information. TR-CSE-93-6, Dept. of Computer Science and Engineering, Florida Atlantic University, 1993.

[9] J. Wu. Fault-tolerant nonredundant broadcasting in hypercubes. *Proc. of 21th International Conference on Parallel Processing.* Sept. 1992, 23-26.

[10] J. Wu and E. B. Fernandez. Reliable broadcasting in faulty hypercube computers. to appear in Microprocessing and Microgramming, 1994.