Fault Tolerant Broadcasting in SIMD Hypercubes

Yeimkuan Chang
Department of Computer Science, Texas A&M University
College Station, Texas 77843-3112
E-mail: ychang@cs.tamu.edu

Abstract – *In this paper, we propose an optimal fault tolerant broadcasting algorithm which requires only $n+1$ steps for an SIMD hypercube with up to $n-1$ faulty nodes. The basic idea of the proposed algorithm is first to find a fault-free subcube (C_S) which contains the source node (S) such that each neighboring subcube of the subcube C_S contains at least one fault. Next, the message is broadcast along the internal dimensions followed by external dimensions of the subcube C_S. This process requires n steps. Since this process does not guarantee that all the fault-free nodes receive the message, an extra step may be needed. We prove that there exists an internal dimension of the subcube C_S such that all the nodes which did not receive the message in n steps will receive the message by broadcasting the message along that dimension. We also develop a generalized broadcasting algorithm which tolerates any number of faults.*

1 Introduction

Hypercube architecture has received much attention for massively parallel processors due to its attractive properties [1]. Some examples of commercial machines based on hypercube are nCUBE/2, iPSC and CM-2. The nCUBE and iPSC are MIMD hypercubes, whereas the CM-2 is an SIMD hypercube. The routing and broadcasting operations in MIMD hypercubes have been extensively addressed in [2]. The fault tolerant routing and broadcasting schemes have been also proposed in [3, 4].

However, very few papers have addressed the fault-tolerant broadcasting in the SIMD hypercube multiprocessors. Since the SIMD hypercubes are the restricted version of the MIMD hypercubes, an efficient SIMD broadcasting algorithm is more difficult to implement than the MIMD counterpart. The simplest algorithm to broadcast in the SIMD n-cube containing up to $n-1$ faulty nodes takes $2n$ steps and requires no knowledge of fault locations [5]. This algorithm forwards messages successively along dimensions 0, 1,..., $n-1$ twice. The best known algorithm proposed in [6] takes $n+12$ steps in an n-cube containing at most $n-1$ faults.

In this paper, we assume that the faulty nodes of the hypercubes are detected and known before the fault-tolerant broadcasting takes place. The links incident on the faulty nodes are also assumed to be faulty. A broadcasting algorithm is said to be optimal if it passes data from the source node to all fault-free nodes in the minimum possible steps. We present an optimal broadcasting algorithm which takes $n+1$ steps in an n-cube containing up to $n-1$ faulty nodes. We also study the situation with n or more faults in an n-cube. The proposed broadcasting algorithm takes $n+7$ and $n+22$ steps for an n-cube containing up to $2n-3$ and $4n-9$ faults, respectively. It is known that even in MIMD hypercubes the broadcasting takes $n+1$ steps in presence of up to $n-1$ faults [7, 4].

The rest of this paper is organized as follows. In Section 2, the optimal fault tolerant broadcasting algorithm is presented. We develop a generalized version of the broadcasting algorithm to handle a large number of faults in Section 3. In Section 4, an efficient algorithm is proposed to find the fault-free subcube containing the source node such that all the neighboring subcubes are faulty. Finally, the concluding remarks are given in Section 5.

2 Optimal Algorithm for $n-1$ faults

The standard SIMD broadcasting algorithm for sending a message to all nodes from a source node uses a dimension sequence which is pre-determined to be 0, .., $n-1$. In each step, only the nodes which have a copy of the message will forward the message to the neighbors. As a result, the message flows along the paths of a binomial spanning tree of the hypercube. Fig.1 shows an example in which the source node is 0000 and dimension sequence of broadcasting in a 4-cube is 0, 1, 2, and then 3. The message is first forwarded to node 0001 along dimension 0, and then to 0011 from 0001 and to 0010 from 0000 along dimension 1 in the next step and so on. The subscripts of $PE's$ shown inside the circles indicate the order of the dimensions in which a message travels to reach the specified $PE's$. After 4 steps, all the nodes receive a copy of the message.

The fault-free SIMD broadcasting algorithm does not work when one or more faults exist since the order of dimensions used for broadcasting is pre-determined. For example, assume node 0001 is faulty in Fig.1. Half the number of nodes in the system, i.e. the nodes in subcube ***1, are blocked from receiving the message if we follow the dimension sequence of fault-free broadcasting algorithm. However, if we redraw the binomial spanning tree for the dimension sequence of 1, 2, 3, and 0, we can see that faulty node 0001 will be located in one of the leaf nodes of the spanning tree. Thus if we broadcast the message along dimensions 1, 2, 3, and then 0, four steps are sufficient to complete the broadcasting in the 4-cube containing only one fault. However, when the number of faults in the system exceeds one, the situation becomes complicated.

We can develop a more general algorithm which is obtained by slightly modifying the fault-free broadcasting algorithm. We use a dimension sequence $[d_1,..,d_{max}]$ to

1063-6374/93 $03.00 © 1993 IEEE

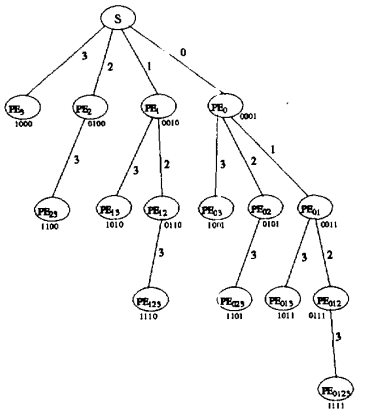

Figure 1: The binomial spanning tree for broadcasting.

successfully broadcast the message from the source node to all the other fault-free nodes. As we shall see, the subject of our paper is to determine max, the number of steps which are needed to guarantee a successful broadcasting, and the ordering of the broadcasting dimensions.

We propose a 4-phase algorithm to broadcast a message from the source node S to all the other fault-free nodes in an n-cube containing up to $n-1$ faulty nodes. The algorithm is outlined as follows.

Algorithm *Broadcast*1(S, n, m), $m \leq n-1$

1. Find the largest fault-free subcube $C_{d,S}$ which is a d-cube containing the source node S. (This phase may be modified later.)
2. Broadcast along the dimensions internal to $C_{d,S}$.
3. Broadcast along the dimensions external to $C_{d,S}$ (This phase is important for n-cube with more than $n-1$ faults and will be discussed later).
4. Select an extra internal dimension such that the pseudo-faulty nodes will receive messages at the $(n+1)^{th}$ step by forwarding messages along the selected dimension.

The number of faulty nodes in the n-cube is assumed to be m which is less than or equal to $n-1$. The internal dimensions of a subcube are the ones which span the subcube. Similarly, the external dimensions of a subcube are the ones which are not internal to the subcube. As we can see, after the first two phases, all the nodes in $C_{d,S}$ receive a copy of the message. Notice that since $C_{d,S}$ is the largest fault-free subcube containing S, each neighboring d-cube of $C_{d,S}$ contains at least one faulty node. At this point, it seems that we need to be careful about determining the order of external dimensions of $C_{d,S}$ for broadcasting since ordering the subsequent broadcasting dimensions determines the number of the fault-free nodes being blocked from receiving messages. However, as we will show later, the ordering of external dimensions is not necessary for the n-cube with up to $n-1$ faulty nodes. Thus the third phase just broadcasts the message along the external dimensions of $C_{d,S}$ in any order. The first three phases require n steps.

However, n steps may not sufficient to broadcast the message from the source node to all the other fault-free nodes. We define the fault-free nodes which do not receive messages after n steps as *pseudo-faulty* nodes. Thus, the last phase is to select a dimension internal to $C_{d,S}$ such that the pseudo-faulty nodes will receive messages in the $(n+1)^{th}$ step.

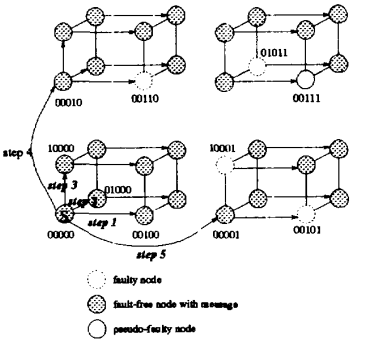

Figure 2: Selection of the extra dimension.

The last phase selects a dimension i internal to $C_{d,S}$ such that for any pair of nodes that are neighbors along dimension i, at least one of them is neither faulty nor pseudo-faulty. We call such dimension i as a *free dimension*. Similarly, a dimension is defined as a *busy dimension* if there exists a pair of faulty or pseudo-faulty nodes along that dimension. Thus, if the messages are broadcast along the free dimension at the $(n+1)^{th}$ step, all the pseudo-faulty nodes will receive messages. For example, Fig.2 shows a 5-cube containing 4 faulty nodes. The largest fault-free subcube containing the source node $S =$ 00000 is $***00$. The dimension sequence for broadcasting is 2, 3, 4, 1, 0. Therefore, the only pseudo-faulty node after 5 steps is node 00111. By forwarding the messages along dimension 2, 3, or 4 at the 6^{th} step, node 00111 will receive a copy of message.

To prove there always exists a free dimension, we partition the n-cube into d-cubes at external dimensions. This results in an $(n-d)$-cube whose nodes are d-cubes (denoted as supernodes). We label each supernode as $PE_{i_0 i_1 .. i_j}$ according to the broadcasting sequence of external dimensions, for $0 \leq j \leq n-d-1, i_j < i_{j+1}$. The relationship between PE's is depicted in the spanning binomial tree of Fig.1. Notice that each circle in Fig.1 represents a supernode and $PE_{i_0 i_1 .. i_j}$ is in the level $j+1$ of the spanning binomial tree.

We define a *transfer* operation in the following manner. For each subtree T_i rooted at PE_i for $0 \leq i < n-d$, the transfer operation moves the faulty nodes in the intermediate and leaf nodes of T_i to PE_i. The transfer operation actually changes the physical locations of the faulty nodes. The busy dimensions in the intermediate and leaf nodes of T_i remain busy after transfer operation since the local addresses of faulty nodes are not changed. Notice that the transfer operation may increase the number of busy dimensions. Thus, if we can find a free dimension for the system with transfer operation which moves the faulty nodes to the root of the corresponding subtree, we can say that there also exists a free dimension without transfer operation.

Let f_i be the number of faults in the subtree T_i, where

$1 \leq f_i \leq n-d$. After n steps, there are $(d-f_i+1)$ free dimensions in subtree T_i according to the result in [8] which states that, for all $n \geq 1$, given any set of $n-k$ or fewer faulty nodes in an n-cube, there exist $k+1$ free dimensions. Equivalently, there are (f_i-1) busy dimensions. Totally, there are $\sum_{i=0}^{i=n-d-1}(f_i-1) = d-1$ busy dimensions since $\sum_{i=0}^{i=n-d-1}(f_i) = n-1$. However, there are d dimensions in each supernode. Thus, there must exist a free dimension internal to the supernodes. Then we have the following theorem.

Theorem 1: *Broadcast*1 takes at most $n+1$ steps to broadcast a message from the source node to all fault-free nodes in an n-cube containing up to $n-1$ faults.

3 n or More Faults

In this section, we generalize the proposed optimal algorithm *Broadcast*1 to handle n or more faults. In the worst case, n faults are sufficient to disconnect one fault-free node from other fault-free nodes. Therefore, we assume that not all the neighbors of a fault-free node are faulty in order to avoid disconnecting the fault-free nodes in the system. Nevertheless, the likelihood of having a disconnected component due to n faulty nodes is negligible. We shall see that the above assumption can be relaxed by declaring the disconnected fault-free nodes as faulty without increasing too many faulty nodes.

As mentioned earlier, the broadcasting order of external dimensions is important for the n-cube with more than $n-1$ faults. The goal of ordering the external dimensions for broadcasting is to minimize the number of fault-free nodes being blocked from receiving messages by the faulty nodes. We propose a broadcasting rule to order the external dimensions of $C_{d,S}$ as follows. For each dimension i which is external to $C_{d,S}$, we split the n-cube into two $(n-1)$-cubes, $C^i_{n-1,S}$ which contains the source node and C^i_{n-1} which does not contain the source node. We broadcast along dimension i earlier than dimension j if C^i_{n-1} contains less number of faulty nodes than C^j_{n-1}. If there is a tie, we make a random selection. We shall see that this broadcasting rule arranges the order of the binomial spanning tree in such a way that most of the faulty nodes are in the leaf nodes of the spanning tree. The broadcasting rule gives us better performance on average when the number of faults exceeds $n-1$.

Although the broadcasting rule gives better performance on average, it does not indicate how many steps are needed to guarantee a successful broadcasting. Thus, in the following, we use the divide-and-conquer technique to develop a general broadcasting algorithm to tolerate a large amount of faults. The fundamental part of the general algorithm is *Broadcast*1. The idea is as follows. Given m faults in an n-cube, we first select an $(n-1)$-cube which contains the least number of faults compared with the number of faults in other possible $(n-1)$-cubes in the n-cube. Next, we employ *Broadcast*1 to broadcast messages to all fault-free nodes in the selected $(n-1)$-cube. Then, the messages are forwarded from the selected $(n-1)$-cube to another $(n-1)$-cube. Finally, one or more dimensions are selected to forward messages to pseudo-faulty nodes. Due to space limit, readers may refer to [9] for detail.

We show that the general algorithm can achieve the broadcasting in $n+7$ steps in an n-cube containing at most $2n-3$ faults. It is easy to see that either Q_{n-1} or Q'_{n-1} contains less than or equal to $n-2$ faults since there are at most $2n-3$ faulty nodes in the system. *Broadcast*1 will always succeed since we assume there is no healthy node whose neighbors are all faulty.

Suppose that Q_{n-1} contains up to $n-2$ faults. By *Broadcast*1, it takes $1+(n-1)$ steps to broadcast messages to all the nodes in Q_{n-1}. One more step can be used to broadcast messages from nodes in Q_{n-1} to the corresponding nodes in Q'_{n-1}. At this point, all the fault-free nodes in Q'_{n-1} receive messages except the nodes which have faulty neighbors in Q_{n-1}. Totally, there are $2n-3$ faulty or pseudo-faulty nodes in Q'_{n-1}. Here we need the following lemma.

Lemma 1 [6]: In any n-cube Q_n containing at most $2n$ faults, there exists three dimensions $J = \{j_1, j_2, j_3\}$ such that for any $j \in J$, there are at most three doubly-faulty j-edges.

A doubly-faulty j-edge is an edge across dimension j which has two faulty nodes at two ends of the edge. Thus, there exists one dimension j such that there are at most three doubly-faulty j-edges. In one step, messages can be moved to all the pseudo-faulty nodes except the pseudo-faulty nodes on the doubly-faulty j-edges. In these three doubly-faulty edges, there are at most six pseudo-faulty nodes. Therefore, we can use three steps to forward the message to one pseudo-faulty node of each doubly-faulty j-edges, since each fault-free node has at least one fault-free neighbor. Then we take one step to forward messages to the other pseudo-faulty nodes on the j-edges along dimension j. Thus, it takes $n+6$ steps to broadcast if the number of faults in Q_{n-1} is less than or equal to $n-2$.

If the number of faults in Q_{n-1} is less than or equal to $n-2$. The message is first routed to S' from S. Then S' is treated as a source node. The above process is applied. Thus, it takes $n+7$ steps. Therefore we conclude that, to broadcast a message from a source node in a faulty SIMD hypercube with $2n-3$ faulty nodes, $n+7$ steps is sufficient.

By the same method, we can develop a broadcasting algorithm for the n-cube containing $4n-9$ faults in which $n+22$ steps are taken. The detail of this algorithm and the simulation results for the generalized broadcasting algorithm which handles any number of faults can be found in [9].

4 Efficient determination of $C_{d,S}$

For an n-cube containing any number of faulty nodes, the determination of the maximum fault-free subcube containing a given node is known to be an NP-complete problem [10, 11]. Fortunately, determining the maximum fault-free subcube is not a necessary requirement for achieving the optimal broadcasting in an n-cube containing at most $n-1$ faulty nodes. Remember that the analysis of our optimal broadcasting algorithm is actually based on the fact that each neighboring d-cube of $C_{d,S}$ contains at least one fault. Thus, instead of finding the largest $C_{d,S}$, the first phase of *Broadcast*1 is changed to finding a *prime* d-cube which contains the source node.

The prime subcube is defined as the one whose neighboring subcubes contain at least one fault. We shall see that the largest fault-free subcube containing the source node is a prime subcube, but not vice versa. For example, given the same faulty nodes in a 5-cube shown in Fig.2, the prime subcube could be 000**. Each of its neighboring 2-cubes, i.e. 001**. 010**, and 100**, contains at least one faulty node. As a result, the algorithm *Broadcast*1 can be applied. In this example, seven nodes become pseudo-faulty after 5 steps of broadcasting along dimensions 0, 1, 2, 3, and 4. Since all the internal dimensions are free, either dimension 0 or 1 can be selected as the extra dimension.

An efficient algorithm to find the fault-free prime subcube containing the source node is given as follows. Let F be the set of the n-bit addresses of faulty nodes in an n-cube. We first search for a fault-free 1-cube which contains the source node (S) by checking if there exists a dimension i such that the i^{th} neighbor ($S^{(i)}$) of S is not faulty. If a fault-free 1-cube containing S is found, then we mask the i^{th} bit of S and the i^{th} bits of addresses of the faulty nodes. By ignoring the masked bits, S and the elements in F become $(n-1)$-bit addresses. The above process is repeated for the set F and the source node S with ignorance of the masked bits until the neighbors of S are all faulty. Notice that different order of the selected dimensions results in different prime subcube containing the source node. The time complexity of above algorithm is $O(mn^2)$ which is derived in [9].

For phases 2 and 3 of the algorithm *Broadcast*1 for $m \leq n-1$ faulty nodes, we need no time since any order of the internal and external dimension suffices. However, for $m \geq n$, we need the broadcasting rule for the external dimensions of $C_{d,S}$ to increase the success rate of the broadcasting as proposed in previous section. Since the time complexity for ordering the external dimensions according the proposed broadcasting rule is $O(mn^2)$ since there are m n-bit addresses and there are n dimensions to be searched. The last phase of the broadcasting algorithm which is equal to finding a free dimension takes $O(m^2n)$ sequential steps [5]. Thus the overall sequential time complexity of the broadcasting algorithm is $O(mn^2 + m^2n)$ in an n-cube with m faults.

Consider an all-to-all broadcast situation, different source nodes may need different dimension sequences to be able to broadcast. The algorithm above only forms the dimension sequence for a specific node S. We shall see that all the nodes in $C_{d,S}$ can use the same dimension sequence as that of S. However, we still need to compute the dimension sequence for other nodes not in $C_{d,S}$. If we use the above efficient algorithm to find the prime subcubes for each individual fault-free node, we need $O(n^3 \times 2^n)$ time complexity if $m = O(n)$. It is so inefficient. In the following, we develop an efficient algorithm which computes the dimension sequences for all the fault-free nodes.

The basic idea of the algorithm is to remove the faulty nodes one by one by splitting the n-cube containing m faults into different disjoint fault-free subcubes. First, we split the n-cube around a faulty node into n disjoint subcubes with dimension i for $0 \leq i \leq n-1$. For each subcube containing faults, we recursively split it until there is no faults in the subcubes. The total time complexity to remove all the faulty nodes is $T(n, F) = O(mn)$ since it takes $O(n)$ time units to remove a fault. It is not difficult to see that $T(n, F)$ also indicates the number of disjoint fault-free subcubes. Hence for the nodes in a fault-free subcube, it takes $O(mn^2 + m^2n)$ to determine the internal, external, and extra dimensions. Since there are $O(mn)$ disjoint subcubes the overall time complexity of the algorithm to determine the broadcasting dimension sequence is $O(m^2n^3 + m^3n^2)$.

5 Concluding Remarks

We have proposed an optimal SIMD broadcasting algorithm for an SIMD hypercube containing up to $n-1$ faults. A generalized broadcasting algorithm was also developed to tolerate any number of faults with a very high probability. We show that the generalized broadcasting algorithm takes at most $n+7$ and $n+22$ steps in the n-cube containing $2n-3$ and $4n-9$ faults, respectively.

Acknowledgment

The author wishes to thank Dr. L. N. Bhuyan for his active supervision during the progress of this work.

References

[1] M. Y. Chan and S. J. Lee, "Fault-Tolerant Embedding of Complete Binary Trees in Hypercubes," *IEEE Transactions on Parallel and Distributed Systems*, pp. 277–288, March 1993.

[2] L. Bhuyan and D. Agrawal, "Generalized Hypercube and Hyperbus Structures for a Computer Networks," *IEEE Transactions on Computers*, pp. 323–333, April 1984.

[3] P. Ramanathan and K. G. Shin, "Reliable Broadcast in Hypercube Multiprocessors," *IEEE Transactions on Computers*, pp. 1654–1657, Dec. 1988.

[4] T. C. Lee and J. P. Hayes, "A Fault-Tolerant Communication Scheme for Hypercube Computers," *IEEE Transactions on Computers*, pp. 1242–1256, Oct. 1992.

[5] C. Raghavendra, P. Yang, and S. Tien, "Free Dimensions - An Effective Approach to Achieve Fault Tolerance in Hypercubes," *Proc. of International Symposium on Fault-Tolerant Computing*, pp. 170–177, 1992.

[6] C. S. Raghevendra and M. A. Sridhar, "Broadcasting Algorithm in Faulty SIMD Hypercubes," In *Proc. of International Symposium on Parallel and Distributed Processing*, pp. 4–11, Dec. 1992.

[7] M. S. Chen and K. G. Shin, "Adaptive Fault-Tolerant Routing in Hypercube Multiprocessors," *IEEE Transactions on Computers*, pp. 1406–1416, Dec. 1990.

[8] J. Bruck, R. Cypher, and D. Soroker, "Tolerating Faults in Hypercubes Using Subcube Partitioning," *IEEE Transactions on Computers*, pp. 599–605, May 1992.

[9] Y. Chang and L. N. Bhuyan, "Fault Tolerant Broadcasting Algorithms in SIMD Hypercubes," Technical report, Texas A&M University, 1992.

[10] B. Becker and H. Simon, "How Robust is the n-Cube ?," *Information and Computation*, pp. 162–178, 1988.

[11] H. L. Chen and N. F. Tzeng, "Quick Determination of Subcubes in a Faulty Hypercube," In *Proc. of International Conference on Parallel Processing*, pp. III–338–345, 1992.