数理解析研究所講究録
第754巻 1991年 145-153

Information Disseminating Schemes and Their Fault Tolerance in Hypercubes

Svante Carlsson†, Yoshihide Igarashi††, Kumiko Kanai††,
Andrzej Lingas†, Kinya Miura†† and Ola Petersson†

†Department of Computer Science, Lund University,
S-221 00 Lund, Sweden
††Department of Computer Science, Gunma University
Kiryu, 376 Japan

Abstract

We present schemes for disseminating information in the n-dimensional hypercube with some faulty nodes/edges. If each processor can send a message to t neighbors at each round, and if the number of faulty nodes/edges is k ($k < n$), then this scheme will broadcast information from any source to all destinations within any consecutive $n + \lceil (k+1)/t \rceil$ rounds. We also discuss the case where the number of faulty nodes is not less than n.

1. Introduction

A number of topologies have been proposed for interconnecting processors in large scaled parallel and distributed systems. The hypercube is one of the most popular and effective network architectures due to its topological richness [2][8]. In fact, hypercube systems have been commercially manufactured and sold [1]. Crucial problems on such parallel and distributed systems are communication cost among processors and communication reliability. That is, the increase of the size of hypercube multicomputers has made the routing and broadcasting problems as well as their fault tolerance important issues [6][10].

Data broadcasting is a very fundamental operation in parallel and distributed systems, and has been intensively studied [2][3][4][7][9]. Broadcasting can be accomplished by the date disseminating process in which each processor repeatedly receives, modifies and forwards messages without physical broadcast. In this fashion multiple copies of the message are disseminated in the systems through disjoint paths. By this multiplicity of message

routing, fault tolerance of the systems can be achieved. In this paper we study information disseminating schemes in the hypercubes with faulty processors/links and their fault tolerance. We assume that all processors are synchronized with a global clock. The period for forwarding a message from a processor to its neighbors is called a round. We evaluate a number of rounds needed to broadcast a message from any starting processor to its all destinations.

2. Hypercubes and Problem Statement

An n-dimensional hypercube (n-cube for short) is defined to be the graph with $N = 2^n$ nodes labeled by the 2^n binary numbers from 0 to $2^n - 1$,where there exists an edge between any two nodes if and only if the binary representations of their labels differ by exactly one bit. For convenience, we will number the bits in an address of a node of the n-cube from right to left as 0 to $n - 1$. The Hamming distance between nodes u and v is denoted by $d(u, v)$. For each node s and d $(0 \leq d \leq n - 1)$,$\oplus_d(s)$ denotes the adjacent node of s such that s and $\oplus_d(s)$ differ in the d-th bit (also called the dth-dimension). For example, if $s = 1010$ then $\oplus_0(s) = 1011, \oplus_1(s) = 1000, \oplus_2(s) = 1110$ and $\oplus_3(s) = 0010$. Let $[m]_n$ be m mod n. The direction set of two nodes $u = u_{n-1}u_{n-2}\cdots u_0$ and $w = w_{n-1}w_{n-2}\cdots w_0$ in the n-cube is defined as $H(u, w) = \{i | u_i \neq w_i\}$.

We assume that each node in the n-cube is a processor and that each edge is a communication line connecting two processors located on its extremities. The information dissemination is a process in which each processor repeatedly receives, modifies and forwards messages. All processors are synchronized with a global clock. The n-cube may contain faulty nodes/edges. In this paper we only consider the case where a faulty processor cannot forward messages. That is, we do not here consider the cases where a faulty processor alters information and forwards the wrong messages.

The problem discussed in this paper can be stated as follows: Given an n-cube containing some faulty nodes/edges, we design a broadcasting scheme, and evaluate its necessary number of rounds to complete the broadcasting. The following conditions must be satisfied:

(1) No matter which processor starts broadcasting, the information should reach all N processors in a certain number of rounds if the number of faulty nodes/edges is within a certain limit.

(2) No matter at what round a processor starts broadcasting, the information should reach all N processors in a certain number of rounds if the number of faulty nodes/edges is

within a certain limit.

3. One Direction Outgoing at Each Round

In this section we assume that each processor in the n-cube can forward a message to a direction at each round. Our scheme for disseminating information is for each processor i, $0 \leq i \leq N-1$, to execute the following procedure.

procedure *1-disseminate*(n)
repeat
  for round := 0 **to** $n-1$ **do**
  { each iteration is called one round and $n = \log_2 N$ }
    for each processor u send message
    from u to $\oplus_{round}(u)$ concurrently
forever

The above scheme is a modification of Han-Finkel scheme for disseminating information on a perfectly connected network[4]. When procedure *1-disseminate*(n) is executed in the n-cube, each processor sends a message to one of its n neighbors at each round. We consider throughout the paper the information disseminating problem on the same assumption as given in Han and Finkel's paper [4]. That is, during each round a processor u sends a message that consists of new information stated by u and/or information received by u during the past rounds. We also assume that messages are long enough to hold all the information that must be sent.

Example 1. Let us consider the 3-cube. The destination of each processor at each round is shown in Table 1.

	processor							
round	0	1	2	3	4	5	6	7
0	1	0	3	2	5	4	7	6
1	2	3	0	1	6	7	4	5
2	4	5	6	7	0	1	2	3

Table 1: Dissemination at each round by 1-*disseminate*(3)

If processor 2 wishes to broadcast some information at round 1, it will send it to processor 0. At round 2,processors $\{0,2\}$ will send the information to $\{4,6\}$. At round 0, processors $\{0,2,4,6\}$ will send the information to $\{1,3,5,7\}$, at which point all the 8 processors will have the desired information.

Theorem 1 *If the n-cube contains no faulty nodes/edges, procedure* 1*-disseminate*(n) *will broadcast information from any source node to all the destination nodes within any consecutive n rounds.*

Theorem 2 *If the n-cube contains not more than k faulty nodes/edges and* $k \leq n-1$*, procedure 1-disseminate*(n) *will broadcast information from any faultless source node to all destination nodes within any consecutive* $n+k+1$ *rounds.*

4. Multiple Outgoing at Each Round

In this section we consider the case where each processor can send messages to multiple directions at each round. Our scheme for disseminating information in such a hypercube model is as follows:

procedure *t-disseminate*(n)
{each processor sends a message to t distinct directions at each round,
where t is an integer between 1 and n}
repeat
  for *round* $:= 0$ **to** $n-1$ **do**
    for each processor u send message from u to processors
    $(\oplus_{[round \times t]_n}(u), \oplus_{[round \times t+1]_n}(u), \cdots, \oplus_{[round \times t+t-1]_n}(u))$
    concurrently
forever

Theorem 3 *If the n-cube contains not more than k faulty nodes/edges and* $k \leq n-1$*, procedure t-disseminate*(n) *will broadcast information from any faultless source node to all destination nodes within any consecutive* $n + \lceil (k+1)/t \rceil$ *rounds.*

5. Dissemination in Hypercubes with Many Faulty Nodes

In this section we consider hypercubes with many faulty nodes. For any faultless node v, if all nodes in the n-cube can receive a message from v within a finite number of rounds, then we say that the n-cube is connected. Even if more than n faulty nodes are contained in the n-cube, it may be possible to broadcast information from any faultless node to all destinations. The 4-cube with 8 faulty nodes as shown in Figure 1 is such an example. For this example 4-*disseminate*(4) will broadcast information from any faultless node to all destinations in 7 rounds.

The following problem arose from the above observation: How many rounds are needed to complete broadcasting in the n-cube with faulty nodes, if the n-cube is connected. In other words, we want to know the smallest k or nearly smallest k such that for every pair of nodes u and v in the n-cube, node v cannot receive a message from node u within consecutive k rounds if and only if v cannot receive the message from u forever. This is an interesting combinatorial problem, but seems to be difficult to solve it.

Let $\alpha = (v_0, v_1, \cdots, v_k)$ be a sequence of nodes in the n-cube. Then α is a path in the n-cube if and only if for each i $(0 \leq i \leq k-1)$, $d(v_i, v_{i+1}) = 1$. If α is a

path in the n-cube and satisfies the condition that for each i $(0 \leq i \leq k-2)$ and s $(2 \leq s \leq n-i)$, $d(v_i, v_{i+s}) \geq 2$, then we say that α is permissible. We define $P(n) = max\{|\alpha| \ |\alpha$ is permissible in the n-cube$\}$, where $|\alpha|$ denotes the path length of α. Obviously $P(0) = 0$, $P(1) = 1$, $P(2) = 2$ and $P(3) = 4$.

The following two theorems are immediate.

Theorem 4 *Suppose that information is disseminated by procedure t-disseminate(n) in the n-cube with some faulty nodes and all faultless edges. Then for any pair of nodes u and v, v cannot receive a message from u within $\lceil n/t \rceil P(n)$ rounds after the starting round if and only if v cannot receive the message forever.*

Theorem 5 *Suppose that information is disseminated by procedure n-disseminate(n) in the n-cube. Then there exist a configuration of faulty nodes, and a pair of nodes u and v such that v receives a message from u in $P(n)$ rounds, but v cannot receive the message from u in a number of rounds less than $P(n)$.*

It seems to be difficult to derive a simple formula for computing $P(n)$. By an exhaustive search and using a computer we have obtained the values of $P(n)$ up to $n = 6$. These values and longest paths that are permissible are shown in Table 2. Unfortunately it takes too much time to compute the values of $P(n)$ for $n \geq 7$, and so far we could not compute them.

We show a lower bound and an upper bound on $P(n)$.

Lemma 1 *For any n and d $(1 \leq d \leq n)$, $P(n) \geq (\lfloor P(d)/2 \rfloor + 1)P(n-d) + P(d)$*

Proof. Let $(A_0, \cdots, A_{P(n-d)})$ be a longest permissible path in the $(n-d)$-cube, and let $(B_0, \cdots, B_{P(d)})$ be a longest permissible path in the d-cube, where each A_i $(0 \leq i \leq P(n-d))$ and each B_j $(0 \leq j \leq P(d))$ are expressed in binary. Consider the following path in the n-cube:

$$
\begin{array}{lcl}
A_0B_0, A_1B_0, & \cdots & , A_{P(n-d)}B_0 \\
 & & A_{P(n-d)}B_1 \\
A_0B_2, A_1B_2, & \cdots & , A_{P(n-d)}B_2 \\
A_0B_3 & & \\
A_0B_4, A_1B_4, & \cdots & , A_{P(n-d)}B_4 \\
\vdots & \vdots\vdots\vdots & \vdots \\
A_0B_{P(d)-2}, & \cdots & , A_{P(n-d)}B_{P(n)-2} \\
A_0B_{P(d)-1} & & \\
A_0B_{P(d)}, A_1B_{P(d)}, & \cdots & , A_{P(n-d)}B_{P(d)}.
\end{array}
$$

This path is permissible, and its length is $(\lfloor P(d)/2 \rfloor + 1)P(n-d) + P(d)$. Hence, the inequality in the lemma holds. □

Theorem 6 *For any $n \geq 3$,*

$$
\begin{aligned}
P(n) &\geq 14^{\lfloor n/6 \rfloor} P([n]_6) + 2(14^{\lfloor n/6 \rfloor} - 1) \\
&\geq 2^{0.634n}.
\end{aligned}
$$

Proof. From Lemma 1 and Table 2,

$$\begin{aligned} P(n) &\geq (\lfloor P(6)/2 \rfloor + 1)P(n-6) + P(6) \\ &\geq 14P(n-6) + P(6) = 14P(n-6) + 26 \\ &\geq 14(14P(n-12) + 26) + 26 \\ &\quad \cdots \\ &\geq 14^{\lfloor n/6 \rfloor} P([n]_6) + 2(14^{\lfloor n/6 \rfloor} - 1) \\ &\geq 2^{0.634n} \quad \text{for } n \geq 3. \end{aligned}$$

□

At present we do not know which value for d is optimal to obtain a good lower bound on $P(n)$ by using Lemma 1. Although the lower bound given in Theorem 6 is the best one known, we doubt that $d = 6$ is optimal to use in the method of Lemma 1.

We now derive an upper bound on $P(n)$.

Definition 1 *Let S be a set of permissible paths. S is said to be independent if and only if for all pairs of nodes, v and w that belong to different paths in S, $d(v, w) \geq 2$.*

Note that if all paths in an independent set of permissible paths in a hypercube are of zero length then S is just an independent set of the hypercube in the standerd graph sense[5].

Lemma 2 *No independent set of permissible paths in the 4-cube contains more than 8 nodes.*

Proof. The assertion of the lemma has been proved by exaustive search by a computer program. □

Lemma 3 *Let S be an independent set of permissible paths in the n-cube, $n \geq 2$. For any k-subcube H_k of the n-cube, S restricted to H_k (denoted by $S|H_k$) is also an independent set of permissible paths in H_k.*

Proof. Consider a pair of nodes occuring on paths in $S|H_k$. Note that if $d(v, w) \geq 2$ in the n-cube then $d(v, w) \geq 2$ in H_k. Therefore, if v and w belong to the same path in $S|H_k$ or belong to different paths in S then $d(v, w) \geq 2$ in $S|H_k$ provided that v and w are not adjacent nodes on the same path in $S|H_k$. It remains to consider the case where v and w belong to different paths, say P_v and P_w in $S|H_k$ which are parts of a single path, say P in S. If v and w were adjacent nodes on P then either they could not be in the same subcube H_k or they would be on the same path in $S|H_k$. Both cases yield a contradiction. We conclude that v and w are not adjacent nodes on P. Hence, $d(v, w) \geq 2$ in the n-cube, and consequently in H_k. □

Theorem 7 *No independent set of permissible paths in the n-cube, $n \geq 4$, contains more than 2^{n-1} nodes.*

Proof. The proof is by induction on $n \geq 4$. For $n = 4$ it follows from Lemma 2. Inductively assume that the assertion of the theorem holds for k satisfyiing $n > k \geq 4$. Let S be an independent set of permissible paths in the n-cube. Arbitrarily split the n-cube into $(n-1)$-subcubes H_{n-1} and H'_{n-1}. If S contains more than 2^{n-1} nodes then S restricted to H_{n-1} or S restricted to H'_{n-1} contains more than 2^{n-2} nodes. By Lemma 3 we obtain a contradiction with the inductive hypothesis. □

As a corollary of the above theorem the next theorem is immediate.

Theorem 8 *$P(n) \leq 2^{n-1}$ for $n \geq 4$.*

Before closing this section we give another example that shows the usefulness of $P(n)$. Suppose that procedure *t-disseminate*(n) is executed. If a processor in the n-cube wishes to know which processors are possible to communicate with, the processor may send a message "Please send your location address". Within $2\lceil n/t \rceil P(n)$ rounds after sending the message the processor knows the following fact: It is not possible to communicate with processors at locations from which the processor has not received their replies. If node v wishes to know whether the n-cube is connected, it may send the same message, "Please send your location address". Let S be the set of nodes which sent back their replies to v within $2\lceil n/t \rceil P(n)$ rounds. If $S \cup \{v\} \cup \{u | u$ is a neighbor of a node in $S \cup \{v\}\}$ is equal to the set of all nodes in the n-cube, then the n-cube is connected, and otherwise it is disconnected. In this fashion any faultless node in the n-cube can decide within $2\lceil n/t \rceil P(n)$ rounds whether the n-cube is connected.

6. Conclusions

We have studied information dissemination for each processor capable of sending multiple data at each round in the hypercube with faulty nodes/edges. We have shown the number of rounds needed to broadcast information from each processor to all destinations on such a model.

A number of open questions remain in the case where the number of faulty nodes is not less than the number of dimensions in the hypercube. The evaluation of $P(n)$ plays an important role in those problems. There is a big gap between our lower bound and upper bound on $P(n)$. A more accurate estimation of $P(n)$ will be interesting for further investigating.

References

[1] M.S Alam and R.G. Melhem, "How to use an imcomplete hypercube for fault tolerance",Proceedings of The 1st European Workshop on Hypercubes and Distributed Computers, Rennes, France, pp.329-341 (1989).

[2] P. Fraigniaud, "Performance analysis of broadcasting", Proceedings of The 1st European Workship on Hypercubes and Destributed Computers, Rennes, France, pp.331-327 (1989).

[3] U. Feige, D. Peleg and P. Raghavan, "Randamized broadcast in networks", Proceedings of SIGAL International Symposium on Algorithms, Tokyo, Japan, Lecture Notes in Computer Science, Vol.45, Springer-Verlag, pp.128-138 (1990).

[4] Y. Han and R. Finkel, "An optimal scheme for disseminating information", Proceedings of 1988 International Conference on Prallel Processing, Chicago, Illinois, pp.198-203 (1988).

[5] F. Harary, "Graph Theory", Addison-Wesley, Reading, Mass. (1969).

[6] J. Hastad, T. Leighton and M. Newman, "Fast computation using faulty hypercubes, "Proceedings of The 21st Annual ACM Symposium on Theory of Computing, pp.251-263 (1989)

[7] S. L. Johnsson and Ching-Tien Ho, "Optimal broadcasting and personalized communication in hypercubes", IEEE Trans. on Computers, Vol.38, pp.1249-1268 (1989).

[8] Y. Lan, A-H. Esfahanian and L. M. Ni, "Multicast in hypercube processors", J. of Parallel and Distributed Computing, Vol.8, pp.30-41 (1990).

[9] P. Ramanathan and K. G. Shin, "Reliable broadcasting in hypercube multiprocessors", IEEE Trans. on Computers, Vol.37, pp.1654-1657 (1988).

[10] Y. Saad and M. Schaltz, "Data communication in hypercubes", J. of Parallel and Distributed Computing, Vol.6, pp.115-135 (1989).

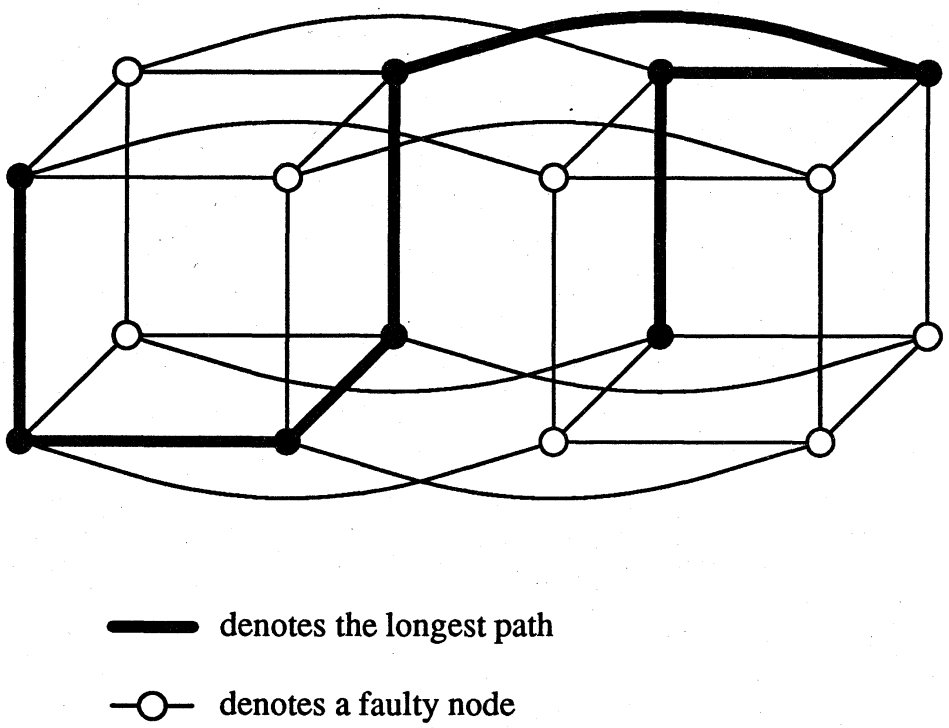

Figure 1. 4-cube with 8 faulty nodes

n = 3	P(3) = 4
path	(000)
	(001)
	(011)
	(111)
	(110)
n = 4	P(4) = 7
path	(0000)
	(0001)
	(0011)
	(0111)
	(0110)
	(1110)
	(1100)
	(1101)
n = 5	P(5) = 13
path	(00000)
	(00001)
	(00011)
	(00111)
	(01111)
	(01110)
	(01100)
	(11100)
	(11101)
	(11001)
	(11011)
	(11010)
	(10010)
	(10110)

n = 6	P(6) = 26
path	(000000)
	(000001)
	(000011)
	(000111)
	(001111)
	(001101)
	(001100)
	(011100)
	(010100)
	(010101)
	(110101)
	(100101)
	(100100)
	(100110)
	(101110)
	(111110)
	(111111)
	(111011)
	(101011)
	(101001)
	(101000)
	(111000)
	(110000)
	(110010)
	(010010)
	(011010)
	(001010)

Table 2: Values of $P(n)$