Message Routing in an Injured Hypercube

Ming-Syan Chen and Kang G. Shin

Real-Time Computing Laboratory
Department of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, MI 48109-2122

ABSTRACT

A connected hypercube containing faulty components (nodes or links) is called an *injured hypercube*. To enable non-faulty nodes to communicate with each other in an injured hypercube, the information of component failures must be made available to those non-faulty nodes for them to route messages around the faulty components.

We develop a fault-tolerant routing scheme which requires each node to know only the information on the failure of its own links. Performance of this scheme is rigorously analyzed. This scheme is not only shown to be capable of routing messages successfully in injured hypercubes when the number of component failures is less than n, but also proved to be able to choose a shortest path with a very high probability.

1. INTRODUCTION

Due to their structural regularity and high potential for the parallel execution of various algorithms, hypercube multicomputers have drawn considerable attention in recent years from both academic and industrial communities [1]. Several research [2] and commercial (by Intel, NCUBE [3], Floating Point System, Ametek, Thinking Machine) hypercube multicomputers have been built, and significant research efforts have been made on hypercube architectures [4, 5, 6, 7, 8].

Efficient routing of messages is a key to the performance of any multiprocessor or multicomputer system. Especially, the increasing use of multiprocessor/multicomputer systems for reliability-critical applications has made it essential to design fault-tolerant routing strategies for such systems. By fault-tolerant routing, we mean the successful routing of messages between any pair of non-faulty nodes in the presence of faulty components (links or nodes) in the system. Several interesting results on the fault-tolerant routing in various networks have been reported [9, 10, 11]. As a partial effort in making hypercube multicomputers attractive for reliability-critical applications, we shall in this paper address the development of fault-tolerant routing schemes for hypercubes. Notice that to route messages in an injured hypercube, we incorporate network information into the message to be transmitted without using additional hardware. This fact distinguishes our work from others, such as [9].

A connected hypercube containing faulty components is called an *injured hypercube*, whereas a hypercube without faulty components is called a *healthy hypercube*. It is well-known that routing in a healthy hypercube can be handled by a systematic procedure [4]. Routing in an incomplete hypercube is also shown to be straightforward [12]. In a healthy hypercube, each intermediate node can determine the next hop of a message by examining the message's destination address and choosing, from all its neighboring nodes, the one which is closest to the destination. Clearly, this can be accomplished by aligning the address of the source node with that of the destination node from right to left and bit by bit [13]. However, this scheme becomes invalid in an injured hypercube, since the message may be routed to the faulty components. In order to enable non-faulty nodes in an injured hypercube to communicate with one another, each node has to be equipped with enough information to route messages to bypass the faulty components. Obviously, it will be very costly for each node to keep and update complete information of the entire hypercube, although, in that case, each node can always find a fault-free path to another node as long as the hypercube remains to be connected. Instead, we would like to incorporate only a small amount of information into each node that is essential for the node to make correct routing decisions.

For the above reasons, we shall develop an adaptive fault-tolerant routing scheme in which each node is required to know only the condition of each link attached to it. This scheme is proven to be capable of routing messages between any pair of non-faulty nodes as long as the total number of faulty components is less than n in a Q_n. More importantly, this scheme, despite of its simplicity, is shown to be very powerful in that the probability of routing messages via shortest paths is very high and the expected length of each routing path is very close to the optimal one.

The paper is organized as follows. Necessary notation and definitions are introduced in Section 2. We shall present in Section 3 an adaptive fault-tolerant routing scheme. Performance of this scheme will be rigorously analyzed in Section 4 in terms of the probability of the shortest path routing and the expected length of each resulting path. This paper concludes with Section 5.

This work was supported in part by the Office of Naval Research under contracts N00014-85-K-0122 and N00014-85-K-0531, and the NASA under grant NAG-1-296. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the view of the funding agencies.

2. PRELIMINARIES

Let $\sum$ be the ternary symbol set {0, 1, *}, where * is a *don't care* symbol, and every subcube in a Q_n can then be uniquely represented by a string of symbols in $\sum$. Such a string of ternary symbols is called the *address* of the corresponding subcube. Fig. 1 shows a Q_2 with address 0*1* in a Q_4. Note that the number of *'s in the address of a subcube is the same as the dimension of that subcube. The rightmost coordinate of the address of a subcube in the n-cube will be referred to as *dimension 1*, and the second rightmost coordinate as *dimension 2*, and so on. For each hypercube node, the communication link in dimension i is also called the *i-th link* of this node. For notational simplicity, we denote the link between nodes 0000 and 0010 as 00-0. Each link is then represented by a "-" symbol in the corresponding binary string.

The *Hamming distance* between two hypercube nodes is the number of bits where their addresses differ from each other. For the nature of distributed routing strategies to be presented, it is necessary to introduce the concept of *relative address* between two hypercube nodes.

Definition 1: The relative address of a node $q = q_n q_{n-1} \cdots q_1$ with respect to another node $m = m_n m_{n-1} \cdots m_1$, denoted by $q_{/m} = r_n r_{n-1} \cdots r_1$, is defined as $r_i = q_i$ if $m_i = 0$ and $r_i = \overline{q_i}$ if $m_i = 1$ for $1 \le i \le n$.

Moreover, the relative address of a subcube with respect to a node u can be determined by the relative addresses (with respect to u) of all the nodes it contains. For example, $0010_{/1001} = 1011$, $0011_{/1001} = 1010$, and $0*1*_{/1001} = 1*1*$. Also, the *spanning subcube* of two hypercube nodes is defined as follows.

Definition 2: The spanning subcube of two nodes $u = u_n u_{n-1} \cdots u_1$ and $w = w_n w_{n-1} \cdots w_1$ in a Q_n, denoted by $SQ(u,w) = s_n s_{n-1} \cdots s_1$, is defined as $s_i = u_i$ if $u_i = w_i$, and $s_i = *$ if $u_i \neq w_i$ for $1 \le i \le n$.

For example, suppose u = 0010 and w = 0111, then H(u,w) = 2 and SQ(u,w) = 0*1*. It is easy to see that SQ(u,w) is the smallest subcube that contains both u and w, and H(u,w) is the dimension of SQ(u,w).

A *path* in a hypercube is represented as a sequence of nodes in which every two consecutive nodes are physically adjacent to each other in the hypercube. The number of links on a path is called the *length* of the path. An *optimal path* is a path whose length is identical to the Hamming distance between the source and destination nodes. Note that due to the special structure of a hypercube, once the source node of a path is given, the path can be described by a *coordinate sequence* that represents the order of the dimensions to be traveled. As shown in Fig. 2, [0001, 0011, 0010, 1010] is an optimal path from the source node 0001 to the destination node 1010, and can also be represented by a coordinate sequence [2, 1, 4]. In the rest of this paper, we shall assume the source and destination nodes are non-faulty and the injured hypercube is connected.

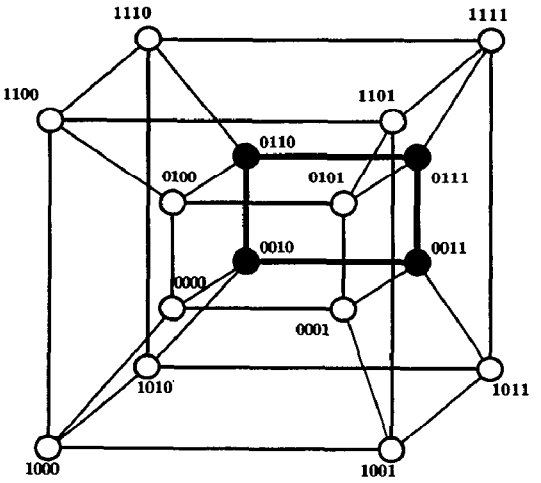

Figure 1. A Q_2 with address 0*1* in a Q_4.

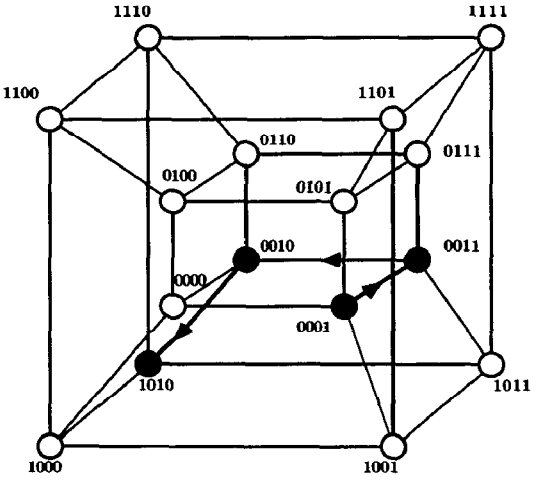

Figure 2. An optimal path from 0001 to 1010.

3. A FAULT TOLERANT ROUTING SCHEME

In this section, we present a fault tolerant routing scheme. Every node in this scheme is required to know only the fault conditions of its own links. As will be shown later, this scheme can successfully route messages between any pair of non-faulty nodes if the number of faulty components is less than n in a Q_n, although the path chosen by this scheme may not always be the *shortest*.

Before we describe the routing algorithm, it is necessary to introduce the following lemma which determines the relative addresses of those nodes in the coordinate sequence of a path.

Lemma 1: Let $[c_1, c_2, \cdots, c_k]$ be the coordinate sequence of a path in a Q_n starting from node u, and $w_{/u} = w_n w_{n-1} \cdots w_1$ denote the relative address of node w with respect to u. Then, the path specified by $[c_1, c_2, \cdots, c_k]$ ends at w if and only if j appears in $[c_1, c_2, \cdots, c_k]$ an even (odd) number of times when $w_j = 0$ (1), for $1 \le j \le n$.

Proof: Traversal of a message along the i-th dimension is the same as inverting the bit in the i-th coordinate of the relative address of its destination. Therefore, traveling along a certain dimension an even number of times has the same effect as not traveling along that dimension at all. Since $w_{/u}$ is the relative address of w with respect to u, this lemma follows. **Q.E.D.**

For example, a path with the coordinate sequence [3, 4, 2] from 0110 will traverse nodes 0010, 1010 and then 1000. When a link becomes faulty, the information about this faulty link can be broadcast to all the other nodes in the hypercube. Several algorithms for accomplishing this were proposed in [11,14]. If each node is equipped with the entire information of the whole network, then Lemma 1 can be used to determine the coordinate sequence of a path which is free of faulty components. However, it is usually too costly (in space and time) to equip each node with information on all failures in the entire network. Instead, in the following subsection we shall develop a fault-tolerant routing scheme which requires each node to know only the condition of its own links.

3.1. Description of Algorithm

To indicate the destination of a message, the coordinate sequence of a path is sent along with the message. Besides, each message is accompanied with an n-bit vector $tag = d_n d_{n-1} \cdots d_1$ which keeps track of "spare dimensions" that are used to bypass faulty components. All bits in the tag are reset to zeros when the source node begins routing of a message. Therefore, such a message can be represented as (k, $[c_1, c_2, \cdots, c_k]$, message, tag), where k is the length of the remaining portion of the path and updated as the message travels towards the destination. A message reaches its destination when k becomes zero.

When a node receives a message, it will check the value of k to see if the node is the destination of the message. If not, the node will try to send the message along one of those dimensions specified in the remaining coordinate sequence. (Note that the coordinate sequence will also be updated as the message travels through the hypercube.) Each node will, of course, attempt to route messages via shortest paths first. However, if all the links in those dimensions leading to shortest paths are faulty, the node will use the spare dimension to route the message via an alternative path. (Recall that the spare dimension is kept track of by the tag.) More formally, this routing scheme can be described in an algorithm form as follows.

Algorithm A: An adaptive fault-tolerant routing.
{Each node receiving (k, $[c_1, c_2, \cdots, c_k]$, message, tag) must execute the following.}
if k=0 **then** {the destination is reached!}
else begin
for j := 1, k **do**
if (the c_j-th link is not faulty) **then**
begin
send ($k-1,[c_1, \cdots c_{j-1}, c_{j+1}, \cdots, c_k]$,message,tag) along the c_j-th link;
stop; {terminate Algorithm A}
end_begin
end_do

{If the algorithm is not terminated yet, all dimensions in the coordinate sequence are blocked because of faulty components.}
for j := 1, k **do** {record all blocked dimensions in *tag*.}
$d_{c_j} := 1$
end_do;

h := min {i : $d_i = 0$, $1 \le i \le n$}; {choose a spare dimension}
$d_h := 1$;
send (k+1, $[c_1, c_2 \cdots, c_k, h]$, message, tag) along the *h-th* link;
stop; {terminate Algorithm A}
end_begin

Consider the Q_4 in Fig. 3, where links 0-01, 1-01 and 100- are faulty. Suppose a message, fm, is routed from u = 0110 to w = 1001. The original message in u = 0110 is (4, [1,2,3,4], fm, 0000). Following the execution of Algorithm A, the node u sends (3, [2,3,4], fm, 0000) to node 0111 which then sends (2, [3,4], fm, 0000) to node 0101. Since the 3-*rd* dimensional link of 0101 is faulty, 0101 will route (1, [3], fm, 0000) to 1101. However, since the 3-*rd* dimensional link of 1101 is faulty, 1101 will use the 1-*st* dimension (tag = 0100 then), and send (2, [3,1], fm, 0101) to 1100, which will, in turn, send (1, [1], fm, 0101) to 1000. Again, the first link of node 1000 is faulty. The 2-*nd* dimension (tag= 0101 then) will be used and (2, [1,2], fm, 0111) is routed to 1010. After this, the message will reach the destination 1001 via 1011. The length of the resulting path is 8.

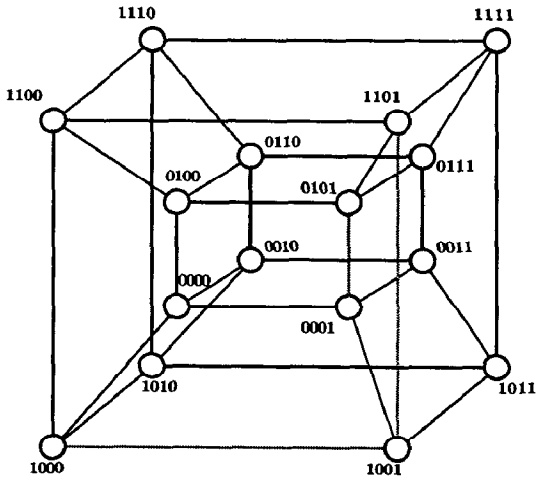

Figure 3. An injured Q_4 where links 0-01, 1-01 and 100- are faulty.

4. Analysis of Algorithm A

As it will be proved below, Algorithm A can route messages between any two non-faulty nodes if the number of faulty components is less than n.

Theorem 2: Algorithm A can always route messages between any two non-faulty nodes successfully as long as the number of faulty components is less than n, i.e., f + g < n, where f and g are the numbers of faulty links and faulty nodes in a Q_n, respectively.

Proof: Note that each node will use a spare dimension only when faulty components are encountered in all the dimensions specified by the coordinate sequence. Call the first node which is forced to use a spare dimension the *obstructed node*. For example, the obstructed node in the example of Fig. 3 is 1101. A faulty component is said to be *type A* if it is (i) adjacent to the obstructed node, say x, and (ii) within SQ(x,w) where w is the destination node. A faulty component is said to be *type B* if it is the first faulty component encountered after the message left the obstructed node. For the example routing in Fig. 3, 1-01 is a type A blocking component and 100- is a type B blocking component, whereas the faulty link 0-01 is neither type A nor type B. Therefore, it can be seen that in the route determined by A, the number of blocking components (including type A and type B) may increase as the message travels towards its destination.

Let b_h denote the blocking component which is encountered first after using a new spare dimension h. Consider two possible cases of b_h: (i) b_h is a link of the destination node, and (ii) b_h is not a link of the destination node. In the case of (i), b_h is the h-th link of the destination node. Since d_h was 0 before the spare dimension h is used, this faulty link had definitely not been encountered before. In the case of (ii), since b_h is the blocking component encountered first after using the spare dimension h, b_h and the set of previous blocking components are placed in the two different Q_{n-1}'s separated by the dimension h. Therefore, from both cases we know the blocking component b_h does not belong to the set of those blocking components that had already been encountered before. Since there are n spare dimensions, we need at least n blocking components to fail the routing. In other words, the routing scheme based on A will never fail as long as the number of faulty components is less than n. This theorem thus follows. **Q.E.D.**

From Theorem 2 and the fact that the number of hops is increased by two whenever a spare dimension is used, the length of the resulting path is H(u,w) + 2k if k spare dimensions are used for routing a message from node u to node w by A. Also, it can be easily verified that the worst case of A needs H(u,w) + 2(n−1) steps to send a message from u to w. In addition, to analyze the performance of A further, we need the following lemma.

Lemma 2: Suppose there are f faulty links in a Q_n, and a message is sent from node u to node w where H(u,w) = k. Let m_A be the Hamming distance between the obstructed node and the destination node. Also, assume Algorithm A is used for routing messages. Then, $P(m_A = j) \le C_{f-j}^{L-j}/C_f^L$ if $1 \le j \le k$, and $P(m_A = j) = 0$ if $j > k$, where $L = n2^{n-1}$ is the number of links in a Q_n.

Proof: It can be seen that the case $m_A > k$ is not possible, because according to A a message is approaching the destination before encountering the obstructed node. Therefore, $P(m_A > k) = 0$.

Consider the case when $1 \le j \le k$. An injured Q_n with f faulty links is called a *configuration*. There are C_{n-1}^L different configurations where $L = n2^{n-1}$. Without loss of generality, we can let $u = 0^n$ and $w = 0^{n-k}1^k$. The problem of obtaining $P(m_A = j)$ is then reduced to that of counting the number of configurations which lead to the case of $m_A = j$. We claim that the number of such configurations is less than or equal to C_{f-j}^{L-j}.

When $m_A = j$, the obstructed node must be within the subcube $0^{n-k}*^k$, and all of its j links towards w must be faulty (j type A blocking components). Although there are many possible locations of the obstructed node, according to the systematic procedure of A, the location of the obstructed node is determined by those non-type-A faulty links which are not within $0^{n-k}*^k$. Suppose $x = 0^{n-k+j}1^{k-j}$ is the obstructed node, then the locations of those j type A blocking components are determined, meaning that there are C_{f-j}^{L-j} different distributions of those non-type-A faulty links. In the case that the distribution of those non-type-A faulty links causes node y, instead of x, to be the obstructed node, we exchange the links (including faulty links) in SQ(y,w) with those in SQ(x,w), and obtain a configuration which leads to the case of $m_A = j$. Notice that some of those C_{f-j}^{L-j} different distributions of non-type-A faulty links may result in $m_A > j$, meaning that the number of configurations leading to $m_A = j$ is less than or equal to C_{f-j}^{L-j}. This lemma thus follows. **Q.E.D.**

From Lemma 2, we can obtain the following theorem which shows that A can route a message to its destination via an optimal path with a very high probability in the presence of link failures.

Theorem 3: Suppose there are f faulty links in a Q_n, and a message is sent from node u to node w where H(u,w) = k. Algorithm A will route a message to its destination via an optimal path (i.e., a path of length k) with a probability greater than $1 - \sum_{j=1}^{k} C_{f-j}^{L-j}/C_f^L$, where $L = n2^{n-1}$.

Proof: From Lemma 2, the probability that A has to use spare dimensions is $\sum_{j=1}^{k} P(m_A = j) \le \sum_{j=1}^{k} C_{f-j}^{L-j}/C_f^L$. Thus, the probability that A will not use spare dimensions is $1 - \sum_{j=1}^{k} P(m_A = j) \ge 1 - \sum_{j=1}^{k} C_{f-j}^{L-j}/C_f^L$. **Q.E.D.**

The following corollary can be derived from Theorem 3.

Corollary 3.1: Suppose there are n−1 faulty links in a Q_n, and a message is sent from node u to node w where H(u,w)=k. Algorithm A will route a message to the destination via an optimal path with a probability greater than $1 - \frac{r_1(1 - r_1^k)}{(1-r_1)}$, where $r_1 = \frac{n-1}{n2^{n-1}}$.

Proof: From Theorem 3, we have

$$\sum_{j=1}^{k} \frac{C_{n-1-j}^{L-j}}{C_{n-1}^{L}} = \frac{n-1}{L} + \cdots + \frac{(n-1)(n-2)\cdots(n-k)}{L(L-1)\cdots(L-k+1)}.$$

Notice that $r_1 = \frac{n-1}{L} > \frac{n-2}{L-1} > \cdots \frac{n-k}{L-k+1}$. Therefore,

$$\sum_{j=1}^{k} \frac{C_{n-1-j}^{L-j}}{C_{n-1}^{L}} < r_1 + r_1^2 + \cdots + r_1^k = \frac{r_1(1 - r_1^k)}{(1-r_1)},$$

$$\Rightarrow 1 - \sum_{j=1}^{k} \frac{C_{n-1-j}^{L-j}}{C_{n-1}^{L}} > 1 - \frac{r_1(1 - r_1^k)}{(1-r_1)}.$$

This corollary thus follows. **Q.E.D.**

Similarly to the above, the performance of A can be analyzed in terms of node failures as follows.

Lemma 3: Suppose there are g faulty nodes in a Q_n, and a message is sent from node u to node w, where H(u,w) = k. Let m_B be the Hamming distance between the obstructed node and

the destination node. Then, $P(m_B = j) \le C_{g-j}^{N-3-j}/C_g^{N-2}$ if $2 \le j \le k$, and $P(m_B = j) = 0$ if $j = 1$ or $j > k$, where $N = 2^n$ is the number of nodes in a Q_n.

From Lemma 3 and the reasoning in the proof of Theorem 3, we can obtain Theorem 4 and its corollary. These state that Algorithm A can also route a message to its destination via an optimal path with a rather high probability in the presence of node failures.

Theorem 4: Suppose there are g faulty nodes in a Q_n, and a message is sent from node u to node w where $H(u,w) = k$. Algorithm A will route the message from u to w via an optimal path of length k with a probability greater than $1 - \sum_{j=2}^{k} C_{g-j}^{N-3-j}/C_g^{N-2}$.

Corollary 4.1: Suppose there are n−1 faulty nodes in a Q_n, and a message is sent from node u to node w where $H(u,w) = k$. Algorithm A will route a message to its destination via an optimal path with a probability greater than $1 - \frac{(n-1)r_2(1 - r_2^{k-1})}{(2^n-2)(1-r_2)}$, where $r_2 = \frac{n-2}{2^n-3}$.

Furthermore, as it will be shown below, the expected length of a path resulting from A is very close to that of the optimal path, i.e., the Hamming distance between the source and destination nodes. Before we proceed, we need to introduce the following proposition.

Proposition 1: Let $\{p_i\}_{i=1}^{n}$ and $\{q_i\}_{i=1}^{n}$ be respectively two decreasing sequences with $p_n = q_n = 0$. Suppose $p_i \le q_i$, for $1 \le i \le n-1$, then $\sum_{i=1}^{n-1} i(p_i - p_{i+1}) \le \sum_{i=1}^{n-1} i(q_i - q_{i+1})$.

Theorem 5: Suppose a message is routed from node u to node w in a Q_n which consists of n−1 faulty links where $H(u,w) = n$. Let H_1 be the length of a path resulting from A and $\Delta H_1 = H_1 - n$. Then, $E(\Delta H_1) \le \frac{n-1}{2^{n-2}}$.

Proof: Notice that $P(\Delta H_1 \ge 2i) \le \sum_{j=1}^{n-i} P(m_A = j)$. Then,

$$E(\Delta H_1) = \sum_{i=1}^{n-1} 2iP(\Delta H_1 = 2i)$$

$$= \sum_{i=1}^{n-1} 2i[P(\Delta H_1 \ge 2i) - P(\Delta H_1 \ge 2(i+1))]$$

$$\le \sum_{i=1}^{n-1} 2i[\sum_{j=1}^{n-i} P(m_A = j) - \sum_{j=1}^{n-i-1} P(m_A = j)]$$

$$= \sum_{i=1}^{n-1} 2iP(m_A = n-i) \quad \text{(By Proposition 1.)}$$

$$\le \sum_{i=1}^{n-1} 2(n-i)C_{n-1-i}^{L-i}/C_{n-1}^{L} \quad \text{(By Lemma 2.)}$$

$$< \sum_{i=1}^{n-1} 2(n-i)r_1^i = 2n\sum_{i=1}^{n-1} r_1^i - 2\sum_{i=1}^{n-1} ir_1^i \quad (r_1 = \frac{n-1}{L})$$

$$= 2nr_1 + 2nr_1(r_1 + \cdots + r_1^{n-2})$$

$$- 2(r_1 + \cdots + (n-1)r_1^{n-1})$$

$$< 2nr_1 = \frac{n-1}{2^{n-2}}. \quad \text{(Since } nr_1 < 1.\text{)} \quad \textbf{Q.E.D.}$$

Also, from Lemma 3 and the reasoning in the proof of Theorem 5, we can obtain the following corollary for the expected length of a path chosen by A in the presence of node failures.

Corollary 5.1: Suppose a message is routed from node u to node w in a Q_n which consists of n−1 faulty nodes where $H(u,w) = n$. Let H_2 be the length of a path resulting from A and $\Delta H_2 = H_2 - n$. Then, $E(\Delta H_2) \le \frac{2n(n-1)(n-2)}{(2^n-2)(2^n-3)}$.

From Theorem 5 and its corollary, it can be verified that the expected length of a path chosen by A is very close to that of the optimal path. Note that due to the absence of global information, the resulting path may not, albeit rare, be the shortest. Clearly, the efficiency of routing can be improved if every non-faulty node is equipped with some more information in addition to those on its own links, since in that case those faulty components on its way to the destination can be foreseen, and thus those faulty components can be bypassed. This, however, will require additional overhead in collecting and maintaining the global information.

5. CONCLUSION

In this paper, we have proposed a distributed fault-tolerant routing scheme for an injured hypercube multicomputer. This scheme is developed in light of the topology of a hypercube and intended to fully use its abundant connections. Performance of this scheme has been rigorously analyzed. We showed that this scheme is not only capable of routing messages successfully in an injured Q_n when the number of component failures is less than n, but also able to choose a shortest path with a very high probability.

REFERENCES

[1] P. Wiley, "A Parallel Architecture Comes of Age at Last," *IEEE Spetrum*, pp. 46-50, Jun. 1987.

[2] C. L. Seitz, "The Cosmic Cube," *Commun. of the Assoc. Comp. Mach.*, vol. 28, no. 1, pp. 22-33, Jan. 1985.

[3] NCUBE Corp., "NCUBE/ten: an overview", Beaverton, OR., Nov. 1985.

[4] Y. Saad and M. H. Schultz, *Data Communication in Hypercubes.* Dep. Comput. Sci., Yale Univ. Res. Rep. 428/85., 1985.

[5] T. F. Chan and Y. Saad, "Multigrid Algorithms on the Hypercube Multiprocessor," *IEEE Trans. on Comput.*, vol. C-35, no. 11, pp. 969-977, Nov. 1986.

[6] L. G. Valiant, "A Scheme for Fast Parallel Communication," *SIAM J. on Computing*, vol. 11, no. 2, pp. 350-361, May, 1982.

[7] B. Becker and H. U. Simon, "How Robust is the n-Cube?," *Proc. 27-th Annual Symposium on Foundations of Computer Science*, pp. 283-291, Oct. 1986.

[8] M. S. Chen and K. G. Shin, "Processor Allocation in an N-Cube Multiprocessor Using Gray Codes," *IEEE Trans. on Comput.*, vol. C-36, no. 12, pp. 1396-1407, Dec. 1987.

[9] E. Chow, H. S. Madan, and J. C. Peterson, "An Adaptive Message-Routing Network for the Hypercube Computer," *Proc. of the Third Conf. on Hypercube Concurrent Computers and Applications*, Jan. 19-20, 1988.

[10] D. K. Pradhan, "Fault-Tolerant Multiprocessor Link and Bus Network Architectures," *IEEE Trans. on Comput.*, vol. C-34, no. 1, pp. 33-45, Jan. 1985.

[11] J. G. Kuhl and S. M. Reddy, "Distributed Fault Tolerance for Large Multiprocessor Systems," *Proc. 7-th Annual Int'l Symposium on Computer Architecture*, pp. 23-30, May 1980.

[12] H. Katseff, "Incomplete Hypercube," *Proc. of Second Hypercube Conf.*, pp. 258-264, Oct. 1986.

[13] C. T. Ho and S. L. Johnsson, "Distributed Routing Algorithms for Broadcasting and Personalized Communication in Hypercubes," *Proc. Int'l Conf. on Parallel Processing*, pp. 640-648, Aug. 1986.

[14] J. R. Armstrong and F. G. Gray, "Fault Diagnosis in a Boolean n Cube Array of Microprocessors," *IEEE Trans. on Comp.*, vol. C-30, no. 8, pp. 587-590, Aug. 1981.