Fault-Tolerant Routing with Regularity Restoration in Boolean n-Cube Interconnection Networks*

Ming-yun Horng and Leonard Kleinrock
Computer Science Department
University of California, Los Angeles
Los Angeles, CA 90024

Abstract

This paper proposes a set of techniques to restore the regularity of a Boolean n-cube network in the presence of node failures, and algorithms to effectively route messages among the surviving nodes. An analytical model to evaluate the degradation of a damaged network is also presented.

One way to restore the regularity of a damaged Boolean n-cube network is by simply disabling the nodes with more than one bad neighbor. The remaining network is called a "1-degraded subnet." A very simple optimal-path routing algorithm, which requires each node to know only its neighbor's status, is developed for such a subnet. Since many nonfaulty nodes may have to be disabled in constructing a 1-degraded subnet, we further develop a heuristic algorithm to restore the network's regularity by constructing a "subnet connected with optimal paths (SCOP)," where only a few nodes must be disabled. The routing algorithm used in 1-degraded subnets also works for SCOPs. To preserve the processing power of the network, we also propose a two-level hierarchical fault-tolerant routing scheme without disabling any nodes.

1 Introduction

A major problem in designing a multiprocessor system is to construct a reliable interconnection network which provides efficient routing of messages among processors. Recently, the Boolean n-cube network (also known as the hypercube network) has become a widely accepted interconnection architecture due to its topological properties as discussed, for example, in [1]. Several research and commercial systems built on this type of interconnection are now available [2].

*This work was supported by the Defense Advanced Research Projects Agency under Contract MDA 903-87-C0663, Parallel Systems Laboratory.

The success of the simple routing algorithms [3] used in Boolean n-cube networks is based on the networks' regularity properties. Although an interconnection network is usually operated in a well-protected environment, faults may occur. When some nodes or communication links fail, the regularity of this "damaged" network is destroyed and the routing algorithm may no longer be applicable. To build a reliable multiprocessor system, the presence of fault-tolerant routing to ensure successful communications between any pair of nonfaulty nodes is essential. Moreover, since the channel speed of an interconnection network is very high, the amount of time that a node can afford to spend in making routing decisions is severely constrained. It is important to have the routing algorithm as simple as possible and hardware realizable.

To successfully route messages in a damaged Boolean n-cube network, either the surviving nodes or the messages must be equipped with information about the locations of the faults. Several algorithms requiring each node of the network to know only the status of its local components (links and nodes) have been presented in [4, 5]. However, the limitation of these approaches is that either the total number of faulty components is very restricted (e.g. less than n) or the number of hops traversed by a message may grow without bound. These problems can be solved by providing each node with more information and having it compute a "safe" route for each message. Algorithms that require each node to know the global status of the network have been reported in [6]. One can even assume the surviving part of the network has an arbitrary topology, in which case each node maintains a routing table as used in networks such as the ARPANET [7, 8]. However, as the network grows in size, the amount of storage space and time needed to maintain and update these routing tables become prohibitive [9].

Since a number of faults in a richly-connected Boolean n-cube network may not destroy its entire regularity, the routing algorithm may take advantage of

0-8186-2310-1/91 $1.00 © 1991 IEEE

the remaining topological regularity. Chen and Shin [10] developed and analyzed a set of fault-tolerant routing algorithms based on the depth-first search principle. In their algorithms, each message contains a tag to keep track of the path traveled so far to avoid visiting a node more than once. To tolerate more than $n-1$ faults, a more complicated procedure is required to guide backtracking whenever a message reaches a dead end. Thus, the length of the packets is variable and the computation overhead is not trivial. To further guarantee that every message is routed to its destination via a shortest path, every node must be equipped with nonlocal status [11].

In this paper, we begin with a queueing model to evaluate the degradation of a damaged Boolean n-cube network with node failures. We then develop a set of techniques to restore the regularity of the network, and algorithms to effectively route messages among the surviving nodes. Our algorithms work under any number of faults as long as the network remains connected.

We restore the regularity of a damaged Boolean n-cube network by disabling the nodes with more than one bad neighbor. The remaining network is called a "1-degraded subnet." We then develop a very simple routing algorithm for such a 1-degarded subnet. With this algorithm, each node only needs to know the status of its neighbors, and every message is routed to its destination via an "optimal path" (to be defined).

Though the 1-degraded subnet can easily be constructed in a distributed manner, many nonfaulty nodes may have to be disabled. We develop a heuristic algorithm to construct a subnet in which every pair of surviving nodes are connected with at least an "optimal" path. In this paper, such a subnet is called a SCOP (Subnet Connected with Optimal Paths). We show that only a small number of nonfaulty nodes will be disabled. The optimal-path routing algorithm for a 1-degraded subnet also works in a SCOP. Some simulation results are presented and compared with a lower bound we develop.

To fully preserve the processing power of the network, we further develop a two-level hierarchical fault-tolerant routing scheme without disabling any nonfaulty nodes. With this approach, a non-optimally connected network is decomposed into a set of clusters such that every cluster forms a subcube with the same property as a SCOP. Each node maintains a small routing table in which every entry of the table corresponds to a destination cluster. Messages are first routed to their destination clusters by use of these routing tables. After a message has arrived at its destination cluster, it is then routed to its destination via an optimal path.

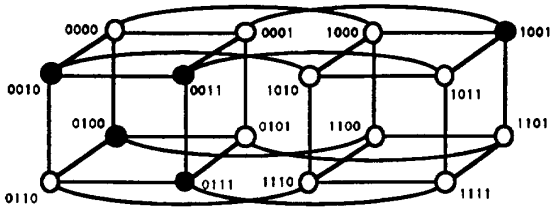

Figure 1: A Boolean 4-cube network with node faults.

2 Preliminaries

A Boolean n-cube network consists of 2^n nodes, each addressed by an n-bit binary number from 0 to 2^n-1. (See Figure 1, where faulty nodes are drawn as black dots.) Nodes are interconnected in such a way that there is a bidirectional link between two nodes, say i and j, if and only if $|i-j|=2^k$ for some integer k from 0 to $n-1$; in this case, we say that these two nodes are linked together in dimension k. For example, in a Boolean 4-cube network, nodes 1000 and 1010 are linked together in dimension 1. It can be seen that by removing all the links in any particular dimension, a Boolean n-cube network is separated into two $(n-1)$-cube networks.

Every "subcube" in a Boolean n-cube network can be uniquely addressed by a string of n symbols drawn from the set {0, 1, X}, where X is a don't care symbol [10]. For example, in a Boolean 4-cube network, nodes 0001, 0011, 0101 and 0111 form a subcube addressed by 0XX1. A node is itself a subcube.

The Hamming distance between any two nodes is defined as the number of bits which differ between their addresses. The length of a path from one node to another is defined as the number of links on the path. An "optimal path" between two nodes is a path whose length is equal to their Hamming distance. A node might not be able to communicate with another via an optimal path in a damaged network. For example, in Figure 1, node 0110 cannot communicate with node 0101 via an optimal path. However, they are able to communicate with each other via the path through nodes 1110, 1100 and 1101. This path is called a shortest path since, among all the possible remaining paths between these two nodes, its length is minimal.

Routing algorithms for Boolean n-cube networks can be found, for example, in [3]. Let the *header* (address portion) of a newly generated message be the exclusive-OR of the message's source and destination addresses. Every one-bit in the header corresponds to a *valid* dimension over which the message can be sent one hop closer to its destination. When a message is sent over a valid dimension, the corresponding one-bit is

changed to zero. Here, the selection of a possible valid dimension for transmission can be adaptive to traffic. A message reaches its destination when its header contains only zeroes. It is clear that, with this algorithm, all messages are routed to their destinations via their optimal paths. However, this routing algorithm cannot work for such a damaged network as shown in Figure 1 since all the optimal paths between nodes 0110 and 0001 are blocked.

In this paper, we make the following assumptions:

- The remaining network is connected.
- Since nodes (or processors) are more complex than links and therefore have higher failure rates, we assume only node failures. To consider a link failure between two nonfaulty nodes, one may disable one of these two nodes. In [12], our algorithms are extended to consider link failures.
- Each node knows the status of its neighboring nodes.
- A node cannot transmit a message to a faulty neighboring node.
- Messages are only destined for nonfaulty nodes.

3 Degradation of Networks with Node Faults

In this section, we present a simple queueing model to evaluate how much a network is degraded by node faults. In many cases it is likely that the failure rate of multiprocessor systems is very small. Let each node fail independently with probability p. We also assume that the arrival of input messages to each node follows a Poisson process with a rate of λ messages per unit time; message lengths are random and drawn independently from an exponential distribution. Since a link survives if and only if both nodes at its ends survive, we have

$$\text{Prob}[A \textit{ link survives.}] = (1-p)^2.$$

Figure 2 shows a Boolean n-cube network with a "cut" in its third dimension. The expected number of surviving links crossing any dimension is given by

$$2^{n-1}(1-p)^2. \tag{1}$$

We further assume that messages are uniformly destined to all other surviving nodes in the network. Thus,

$$\begin{aligned} \alpha &\triangleq \textit{traffic intensity from a given source to a particular destination} \\ &= \frac{\lambda}{2^n(1-p)-1.} \end{aligned}$$

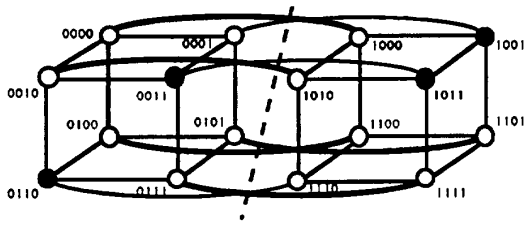

Figure 2: A cut in dimension 3. (Surviving links crossing the cut are shown as heavy lines).

From Figure 2 we note that every message which is generated from a node in the left subcube which is destined to a node in the right subcube must travel over one of the surviving links in order to cross the "cut". If a message travels along an optimal path between these two nodes, the message must travel across the "cut" exactly once. We further assume that the traffic crossing this "cut" is perfectly balanced. The average number of surviving nodes in each subcube is $2^{n-1}(1-p)$. Each will send α units of traffic to every other surviving node in the network. Thus each node will send $\alpha[2^{n-1}(1-p)]$ units of traffic across each cut. Since there are $2^{n-1}(1-p)$ nodes in each subcube doing this, and traffic is balanced on each link, we have

$$\begin{aligned} \rho &= \textit{Traffic load per channel} && (2) \\ &= \frac{[2^{n-1}(1-p)]^2\alpha}{2^{n-1}(1-p)^2} && (3) \\ &= \frac{\lambda}{2(1-p)-2^{1-n}.} && (4) \end{aligned}$$

Here, the channel is unidirectional. Obviously, this model yields an optimistic bound.

We apply Kleinrock's *Independence Assumption* [8] which is often used in the delay analysis of communication networks. This assumption states that each time a message is received at a node within the network, its transmission time is chosen independently from an exponential distribution. We assume the mean transmission time of a message equals one unit of time. Thus, each channel is modeled as an M/M/1 system with Poisson arrivals at a rate $\lambda/[2(1-p)-2^{1-n}]$ and with an exponential service time whose mean is one unit of time. In order for this system to be stable, we require that $\rho < 1$, that is,

$$\lambda < 2(1-p)-2^{1-n}. \tag{5}$$

We define

$$\gamma = \textit{Throughput of the network,}$$

then we have

$$\gamma = \lambda 2^n(1-p) \quad (6)$$
$$< 2[2^n(1-p)-1](1-p), \quad (7)$$

where $2[2^n(1-p)-1](1-p)$ is clearly the mean network communication capacity. The mean message delay is then given by [8]

$$T = \sum_{i=1}^{M} \frac{1}{\gamma} \frac{\rho}{1-\rho}$$
$$= M \left\{ \frac{1}{\lambda 2^n(1-p)} \right\} \left\{ \frac{\frac{\lambda}{2(1-p)-2^{1-n}}}{1-\frac{\lambda}{2(1-p)-2^{1-n}}} \right\}$$

where M is the number of surviving channels in the network ($M = 2^{n-1}(1-p)^2 2n$). Thus,

$$T = \frac{n(1-p)}{2(1-p)-2^{1-n}-\lambda}. \quad (8)$$

We further obtain the following approximations for $n >> 1$.

$$\rho \approx \frac{\lambda}{2(1-p)}, \quad (9)$$

$$\gamma < 2^{n+1}(1-p)^2, \quad (10)$$

and

$$T \approx \frac{n(1-p)}{2(1-p)-\lambda}. \quad (11)$$

In Figure 3 we show the mean message delay obtained from our optimistic model for a Boolean 4-cube network with two different failure rates. We also ran a flow deviation program [14] to find the minimal achievable delay. We find that our assumptions are appropriate if failure rates are small.

Moreover, in most queueing systems, two performance measures, response time and throughput, compete with each other. Typically, by raising the throughput of the system, which is desirable, the mean response time is also raised, which is undesirable. Here, we combine the throughput and the mean message delay of the network into a single measure, *power*, which is defined as follows [13].

$$Power = \frac{Throughput\ of\ the\ network}{Mean\ message\ delay}.$$

A system is said to be operating at an optimal point if the power at that point is maximized. For $n >> 1$, we find that power is maximized when $\lambda = 1-p$ which is equal to half the maximum allowed throughput per node, as found in [13].

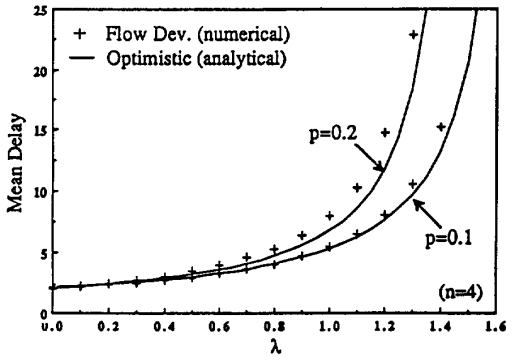

Figure 3: Minimal achievable delay of a Boolean 4-cube network with two different failure rates.

4 Routing in 1-Degraded Subnets

A network is said to be k-degraded if every surviving node in the network has at most k "bad" (to be discussed) neighbors. A damaged network can easily be made k-degraded in a distributed manner as follows: Every surviving node (or nonfaulty node initially) has a list which gives the status of its neighboring nodes. Every surviving node checks its list and disables itself if it has more than k "bad" neighbors; in this case, it must inform all its surviving neighbors of the change in its status. Every surviving node keeps updating its list until it disables itself or the disabling process stops. Here, during each step of the iteration, the "bad" nodes include all faulty nodes and all nodes which have been disabled in previous iterations. We call the remaining network a "k-degraded subnet."

We now present a very simple adaptive routing algorithm for 1-degraded Boolean n-cube subnets. This algorithm requires each node to know only its neighbors' status. We let *neighbor_status* be an n-bit binary number in which a bit is set to one if its corresponding neighbor is surviving. Otherwise, the bit is reset to zero. This routing algorithm is shown in Figure 4, where "&"" is a bit-wise AND function. We note that, with this algorithm, every message is routed to its destination along an optimal path.

The proof that this routing algorithm works for 1-degraded Boolean n-cube subnets is as follows: If a node receives a message with more than one one-bit in its header, the node surely can find a valid dimension (or channel) to transmit the message. If the message has only a single one-bit in its header, then the corresponding neighbor must be surviving. Otherwise, the assumption that messages are only destined for surviving nodes is violated.

```
When a message is received,
if ( header = 0 )
  Send the message to the local processor.
else
  valid_channels <- header & neighbor_status.
  Randomly select a 1-bit from valid_channels.
  Change the selected 1-bit in the header to 0.
  Send the message over the selected channel.
```

Figure 4: The optimal-path routing algorithm for 1-degraded subnets

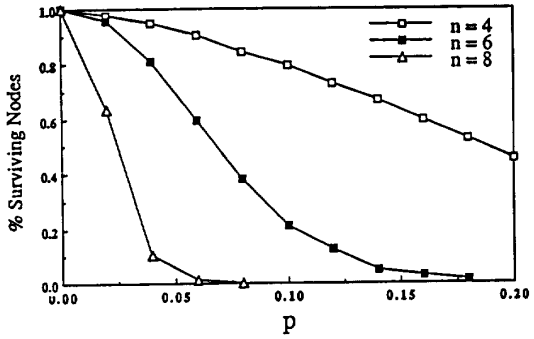

Figure 5: Percentage of surviving nodes in the 1-degraded subnets.

This approach works well in the situation where a whole cluster of nodes has been "bombed out." As an example of such spatially correlated faults, one may consider a power-supply failure which disables the entire cluster of nodes supported by it.

The disadvantage of this approach for 1-degraded subnets is that, in a large Boolean n-cube network with moderate failure rates, since every node has a large number of neighbors, the probability that a node has more than one bad neighbor can be large. As a result, many nodes may have to be disabled; hence, the computational power of the system is significantly reduced. Figure 5 shows, for networks of different sizes, the percentage of nodes that remain after the disabling iteration settles down, given that nodes initially fail independently with a given probability.

5 Construction of a SCOP

In this section, we develop a heuristic algorithm to construct a subnet where every pair of surviving nodes is connected with at least one optimal path (or we say every pair of nodes is *optimally connected*). We call such a subnet a SCOP (Subnet Connected with Optimal Paths). We show that our routing algorithm for 1-degraded subnets works in such a subnet.

In a Boolean n-cube network, a node and its k neighbors can uniquely identify a Boolean k-subcube, where $0 \leq k \leq n$. We note that such a subcube is the smallest subcube containing the node and its k neighbors. For example, in a Boolean 4-cube network, the node 0110 and nodes 0100 and 0111 can identify the subcube 01XX. Moreover, any two nodes which are k hops away in distance can also identify a Boolean k-subcube.

We assume there is a central control unit that collects information from every surviving node of the network and makes decisions about how to disable a node. Here is a heuristic algorithm for constructing a SCOP: We let *List[i]* be a check-list which contains all surviving nodes with i bad neighbors. Again, the "bad" nodes include all faulty nodes and all nodes having been previously disabled. Nodes on *List[i]* have higher priority for disablement than any other nodes on the list with smaller i. That is, the node with most bad neighbors (worst connection) has the highest priority of being disabled.

We choose a node, say node j, from the highest priority non-empty check-list, and simply find the smallest subcube containing node j and all its bad neighbors (e.g. in Figure 1, node 0110 and subcube 0XXX). It is clear that without routing through the links outside the subcube, node j cannot communicate with any other surviving nodes of the subcube. If the total number of surviving nodes in the subcube is more than 2, node j is disabled. If the number of surviving nodes in the subcube is exactly 2, we choose to disable either one of them (See [12].). Otherwise, node j is safe at this moment and is removed from the lists. A safe node may be brought back to the check-lists if any of its neighbors is disabled. The algorithm stops when every surviving node is safe. Figure 6 illustrates a resulting SCOP for the sample Boolean 4-cube network as shown in Figure 1. In [12], we show the number of surviving nodes of the SCOPs obtained from our heuristic algorithm is very close to the number of surviving nodes achievable by an exhaustive search.

We now prove the optimal-path routing algorithm we developed for 1-degraded subnet works for a SCOP. If the destination node of a message is k hops away from the node where the message is currently residing, the current node should have k valid dimensions to choose from. Since the k-subcube identified by the current node and the destination node is optimally connected, the current node must be able to find at least one valid dimension to transmit the message.

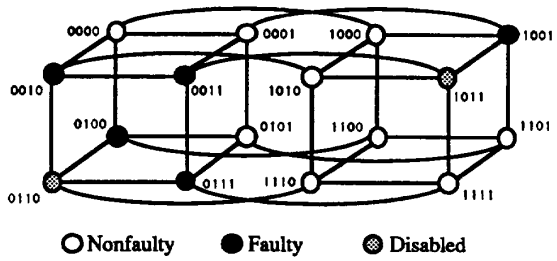

Figure 6: A SCOP of the Boolean 4-cube as shown in Figure 1.

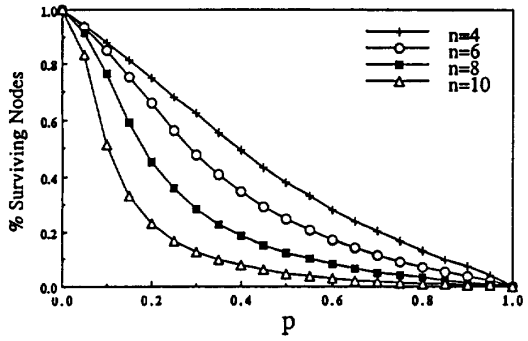

Figure 7: Percentage of nodes remaining in the SCOPs.

Thus, if every subcube is optimally connected, then the optimal-path routing algorithm works.

Figure 7 shows the percentage of nodes which remain in the SCOP, given that nodes initially fail independently with a given probability. Comparing this with the percentage of nodes which remain in the 1-degraded subnet, we find the number of surviving nodes is dramatically increased. In Figure 8, we compare these two disabling schemes by showing the percentage of surviving nodes in a Boolean 8-cube network.

It is very difficult to analytically evaluate the performance of arbitrary networks in a dynamic traffic environment. In this paper, routing in the SCOPs is extensively simulated. To verify the effectiveness of our routing algorithm, for each failure pattern, we also ran a flow deviation program to find the minimal achievable delay. These results are also compared with the optimistic bound obtained in Section 3. Figure 9 shows, for different input rates, the mean message delay in the SCOPs of a Boolean 6-cube network. The results with 95% confidence shown here were from 100 randomly generated patterns, each containing 6 faulty

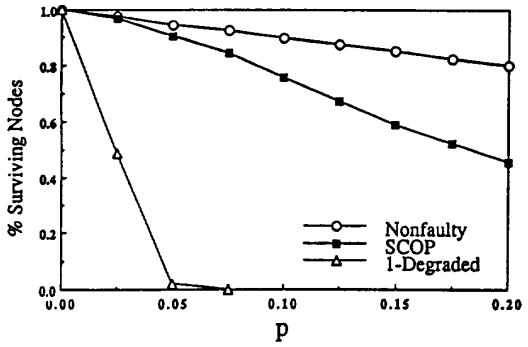

Figure 8: Comparison of percentage of surviving nodes, n=8.

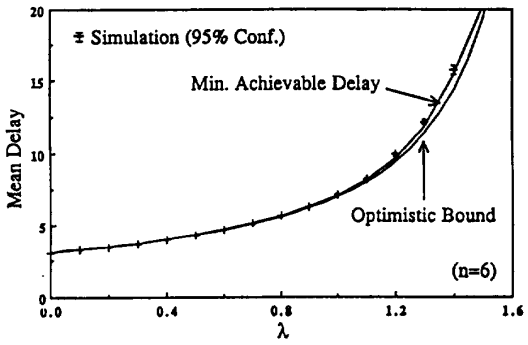

Figure 9: Mean delay of the SCOPs of Boolean 6-cube netowrks. Each network has 6 faulty nodes.

nodes. We find that the mean message delay is very close to the minimal achievable bound, which is also very close to the optimistic bound.

6 Two-level Hierarchical Routing

In this section, without disabling any nonfaulty nodes, we restore the regularity of a damaged Boolean n-cube network by decomposing the network into a set of clusters such that every cluster forms a subcube with the same property of a SCOP (i.e. every pair of the surviving nodes of a cluster is connected with at least an optimal path). A *two-level hierarchical* routing algorithm, which requires every node to maintain a small routing table, is then developed.

node	bad dimensions
0000	1 2
0001	1 3
0101	0 1
0110	0 1 2
1011	1 3

Table 1: Bad dimensions for each surviving node with more than one bad neighbor.

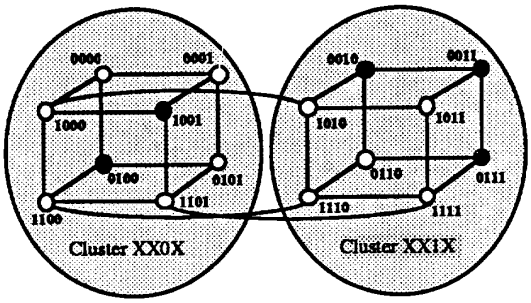

Figure 10: A two-level hierarchical structure of a 4-cube network with 5 node faults as shown in Figure 1.

6.1 The Two-level Network Decomposition

A non-optimally connected Boolean n-cube network is decomposed into a set of clusters as follows. For all the surviving nodes which have more than one bad neighbor, the central control unit counts the number of bad links in each dimension. The central control unit then chooses the dimension having the most bad links and cuts the network into two clusters along this dimension. Each of these two clusters is an $(n-1)$-subcube. A subcube must be further decomposed if not every pair of surviving nodes of the subcube is optimally connected.

Again, as an example, let us examine the Boolean 4-cube network with 5 faulty nodes as shown in Figure 1. Clearly the network is not optimally connected. In Table 1 we show, for each surviving node with more than one bad neighbor, the dimensions along which its neighbors are bad. In this example, dimension 1 has the the maximum number of bad links (i.e. 5). We decompose the network along dimension 1. As a result, the network is separated into the following subcubes: XX0X and XX1X. In this case, both of these two subcubes are optimally connected. The two-level hierarchical structure is shown in Figure 10.

6.2 Routing in the Two-level Hierarchical Network

Messages must first be routed to their destination clusters. Every surviving node maintains a cluster routing table with one entry for each destination cluster. Each entry gives the address of a destination cluster, the best outgoing channel for that cluster, and a relative weight (usually delay is used as the weight). Any algorithm (e.g. the ARPANET-like algorithm) can be used to maintain the cluster routing table.

When a node receives a transit message, if the destination node of the message does not belong to the cluster in which the node resides, the message is sent to a neighbor based on the node's cluster routing table. Otherwise, the message is routed to its destination node using the routing algorithm for 1-degraded subnets. To exploit the possible multiple paths from one node to another in a cluster and balance the network's traffic, the message is sent along a most lightly loaded channel in its destination cluster. We may further improve the performance by providing multipath routing, where each entry of the routing table gives multiple choices of outgoing channels.

6.3 Discussion

This hierarchical routing approach has the following advantages:

- No good links or nodes are eliminated. The processing power of the nonfaulty part of the network is fully maintained.
- The size of the routing table is significantly reduced from the ARPANET-like routing table. In [12], we show that the number of clusters generated by decomposition cannot exceed the number of faulty nodes in the network.
- The optimal-path routing algorithm for 1-degraded subnets works for each cluster.
- The number of hops traversed by a message is bounded.
- The increase in the mean path length caused by hierarchical routing is typically very small [12].

7 Conclusions

In this paper, we first developed a queueing model to evaluate the degradation of a damaged Boolean n-cube network with node faults. We next developed an adaptive fault-tolerant routing algorithm for 1-degraded subnets. This algorithm is very simple; it makes routing decisions based only on the node's local status. This algorithm routes every message to its destination via an optimal path.

We further exploited the remaining regularity of a damaged Boolean n-cube network and developed a heuristic algorithm to construct a subnet (i.e. SCOP) in which every pair of surviving nodes is connected with at least one optimal path. We showed the algorithm used in a 1-degraded subnet also works for a SCOP. Only a small number of nonfaulty nodes must be disabled. The performance of the optimal-path routing algorithm in SCOPs was studied. We found the mean message delay is very close to the minimal achievable bound.

To preserve all the nonfaulty nodes in the network, we also developed a two-level hierarchical routing scheme. A damaged Boolean n-cube network is decomposed into a set of clusters; each of them is a SCOP. Every surviving node in the network is required to maintain a small routing table. A two-level hierarchical routing algorithm has also been developed. This approach maintains the network's rich connection without disabling a single nonfaulty node. In [12], we show that the probability of routing a message to its destination via a shortest path is very high and that the increase in the mean path length caused by hierarchical routing is very small. More simulation results are being collected and some other performance measures such as the mean delay, the throughput of the network and hot spot problems are also being evaluated.

References

[1] Y. Saad and M. H. Schultz, "Topological Properties of Hypercubes," *IEEE Trans. Comput.*, vol. 37, pp. 867-872, July 1988.

[2] J. P. Hayes and T. Mudge "Hypercube Supercomputers," *Proc. of the IEEE*, vol. 27, pp. 1829-1841, Dec. 1989.

[3] H. Sullivan and T. R. Bashkow, "A Large Scale, Homogeneous, Fully Distributed Parallel Machine I," in *Proc. 4th Symp. Comput. Architecture*, pp. 105-117, Mar. 1977.

[4] T.-C. Lee and J. P. Hayes, "Routing and Broadcasting in Faulty Hypercube Computers," in *Proc. 3rd Conf. Hypercube Concurrent Comput. Appl.*, pp. 346-354, Jan. 1988.

[5] J. M. Gordon and Q. F. Stout, "Hypercube Message Routing in the Presence of Faults," in *Proc. 3rd Conf. Hypercube Concurrent Comput. Appl.*, pp. 318-327, Jan. 1988.

[6] M.-S. Chen and K. G. Shin, "On Hypercube Fault-Tolerant Routing Using Global Information," in *Proc. 4th Conf. Hypercube Concurrent Comput. Appl.*, pp. 83-86, Mar. 1989.

[7] J. M. McQuillan and D. C. Walden, "The ARPA Network Design Decisions," *Comput. Networks*, vol. 1, pp. 243-289, Aug. 1977.

[8] L. Kleinrock, *Queueing Systems Volume II: Computer Applications*, John Wiley and Sons, New York, 1976.

[9] L. Kleinrock and F. Kamoun, "Hierarchical Routing for Large Networks, Performance Evaluation and Optimization," *Comput. Networks*, vol. 1, pp. 155-174, Jan. 1977.

[10] M.-S. Chen and K. G. Shin, "Depth-First Search Approach for Fault-Tolerant Routing in Hypercube Multicomputers," *IEEE Trans. Parallel and Distrib. Syst.*, vol. 1, pp.152-159, Apr. 1990.

[11] M.-S. Chen and K. G. Shin, "Adaptive Fault-Tolerant Routing in Hypercube Multicomputers," *IEEE Trans. Comput.*, vol. 39, pp. 1406-1416, Dec. 1990.

[12] M.-Y. Horng, *Performance Analysis of the Boolean n-Cube Interconnection Network for Multiprocessors*, Ph.D. Dissertation, Comput. Sci. Dep., Univ. California, Los Angeles, 1991.

[13] L. Kleinrock, "Power and Deterministic Rules of Thumb for Probabilistic Problems in Computer Communications," *Int. Conf. on Commun.*, pp. 43.1.1-43.1.10, June 1979.

[14] L. Fratta, M. Gerla, and L. Kleinrock, "The Flow Deviation Method - An Approach to Store-and-forward Communication Network Design," *Networks*, vol. 3, pp. 97-133, 1973.