Highly fault-tolerant hypercube multicomputer

B. A. Izadi, F. Özgüner and A. Acan

Abstract: A strongly fault-tolerant design for a d-dimentrun "aucountsional hypercube multiprocessor is presented and its reconfigurability examined. The augmented hypercube has a spare node connected to each node of a subcube of dimension i, and the spare nodes are also connected as a $(d-i)$-dimensional hypercube. By utilising the circuit-switched capabilities of the communication modules of the spare nodes, a large number of faulty nodes and faulty links can be tolerated. Both theoretical and experimental results are presented. Compared with other proposed schemes the approach can tolerate significantly more faulty nodes and faulty links with a low overhead and no performance degradation.

1 Introduction

As the size of the hypercube multicomputer grows, owing to its complexity, the probability of node and/or link failures becomes high. Therefore, it is crucial that such systems be able to withstand a large number of faults. To sustain the same level of performance in the presence of faults, some researchers have investigated hardware schemes for the hypercube where spare nodes and/or links are used to replace the faulty ones. In the literature, a reconfigurable system that retains the same service level, as well as keeping the same system topology after the occurrence of faults, is referred to either as a *strongly fault-tolerant* [1] or a *structurally fault-tolerant* [2] system.

Two classes of hardware schemes have been proposed. Some researchers have examined local reconfiguration techniques where a spare node can only replace a faulty node within a given subset [3–8]. A common drawback of these approaches is low utilisation of spare nodes. Furthermore the schemes do not tolerate any faulty link and generally are not strongly fault-tolerant. The second class of approaches uses a global reconfiguration scheme and is based on creating a supergraph of the hypercube [2, 9–11]. These schemes are strongly fault-tolerant. However, they are mostly node-minimal and suffer from large node degrees.

We propose a global reconfiguration scheme that utilises circuit-switched communication to make the hypercube strongly fault tolerant. In our scheme a d-dimensional hypercube is divided into $2^{(d-i)}$ subcubes of dimension i; we call each of these subcubes a cluster. One spare node is assigned to each cluster; the spare node is connected to every regular node of its cluster via an intracluster spare link. Spare nodes are also interconnected to form a $(d-i)$-dimensional spare cube using intercluster spare links. We call the resultant structure an *enhanced cluster hypercube (ECH)*. Fig. 1 depicts an ECH of dimension $d=4$ with clusters of dimension $i=2$. The spare links are shown using lighter lines. By utilising the circuit-switched communication modules of various spare nodes, edge-disjoint paths between multiple nodes of a cluster and multiple spare nodes can be made. For example, in Fig. 1, using the spare links, nodes 0100, 0110, and 0111 in cluster 01*XX* can be connected to the spare nodes 01*SS*, 00*SS*, and 11*SS*, respectively, without sharing any of the spare links. We use this property to show that multiple faulty nodes in a cluster can be tolerated. Faulty links are bypassed by establishing parallel paths using spare links.

We present both theoretical and experimental results. Our theoretical results, representing the worst-case scenario, indicate that the ECH can tolerate $2^{(d-i)}$ faulty nodes for up to three faulty nodes per cluster; only outline proof of theorems are provided owing to space limitation [12, 13]. Our experimental results, based on random fault distribution of up to $2^{(d-i)}$ faulty nodes, have yielded 100% fault coverage.

2 Notation and definitions

Each node of a d-dimensional hypercube is represented by *d-tuple* $(b_{d-1}\ldots b_i\ldots b_0)$, where $b_i \in \{0, 1\}$. A subcube is defined by a unique *d-tuple* $\{0, 1, X\}^d$ where '0' and '1' are the *bound* coordinates, and 'X' represents the free coordinates. A $(d-k)$-dimensional subcube is represented by a *d-tuple* with k bound coordinates and $(d-k)$ free coordinates. Each spare node is uniquely defined by *d-tuple* $\{0, 1, S\}^d$ where '0' and '1' represent the bound coordinates and S corresponds to the free coordinates of the spare node's assigned cluster. A link is specified uniquely by *d-tuple* $\{0, 1, -\}^d$ where '$-$' can be substituted by '0' or '1' to identify its connecting nodes. An intracluster spare link (a link connecting the spare node to a node within the cluster) is defined by a $(d+1)$*-tuple*$\{0, 1, S\}^{(d+1)}$ where S is inserted to the left of the $(i-1)$th bit of the regular node ID, to which the spare node is connected. For example, the intracluster spare link connecting the spare node 00*SS* and node 0001 is identified as 00*S*01. An intercluster spare link (a link connecting two spare nodes) is defined by a *d-tuple* $\{0, 1, S, -\}^d$ where

---

IEE Proceedings online no. 19990181
DOI: 10.1049/ip-cdt:19990181

Paper first received 19th May 1998, and in revised form 13th Janurary 1999

B.A. Izadi is with Dept. of Electrical and Computer Engineering, State University of New York, 75 South Manheim Blvd., New Paltz, NY 12561
E-mail: bai@engr.newpaltz.edn
F. Özgüner is with Dept. of Electrical Engineering, Ohio State University, 2015 Neil Ave., Columbus, OH 43210, USA
E-mail: ozguner@ce.eng.ohio-state.edu
A. Acan is with the Computer Engineering Dept., Eastern Mediterranean University, Gasi Magosa, TRNC, Mersin 10, Turkey
E-mail: acan@compenet.emu.edu.tr

‘ – ’ can be substituted by ‘0’ or ‘1’ to identify its connecting spare nodes. Hence, the intercluster spare link connecting spare nodes 01*SS* and 00*SS* is labeled 0 – *SS*.

3 Overview of enhanced cluster approach

In an ECH of dimension d and clusters of dimension i, the degree of each regular and spare node is $(d+1)$ and $2^i + (d - i)$, respectively. The block diagrams of a regular node and a spare node for the four-dimensional ECH of Fig. 1 are depicted in Fig. 2. Note that for $i = 1$, the architecture of the regular and the spare nodes would be the same. Hence, one can implement an ECH by using the same node architecture for the spare and regular nodes; the degree of every node would be $(d+1)$.

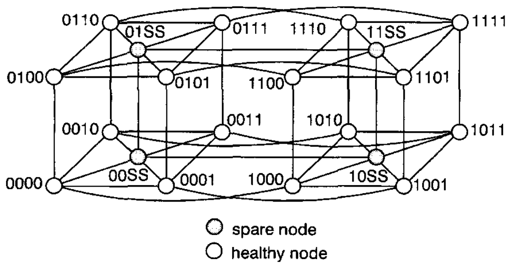

Fig. 1 *Enhanced cluster hypercube of dimension four*

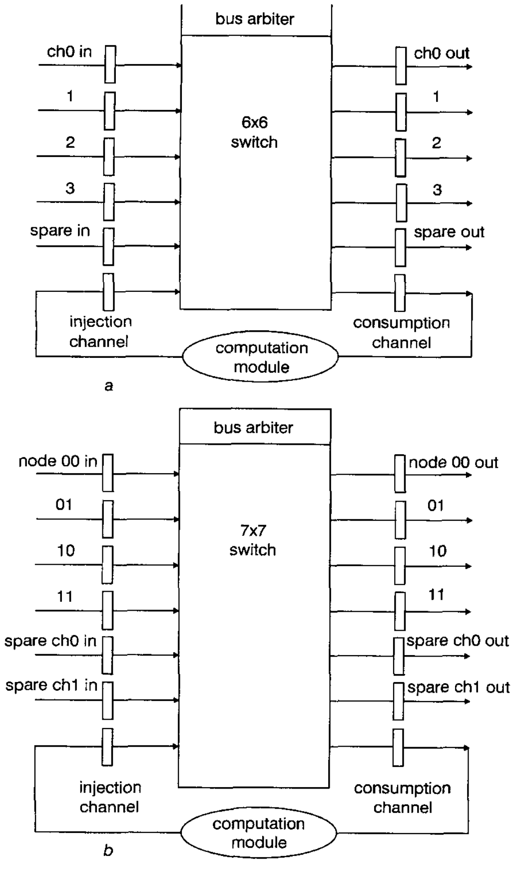

Fig. 2 *Architecture*
(*a*) Regular node (*b*) Spare node

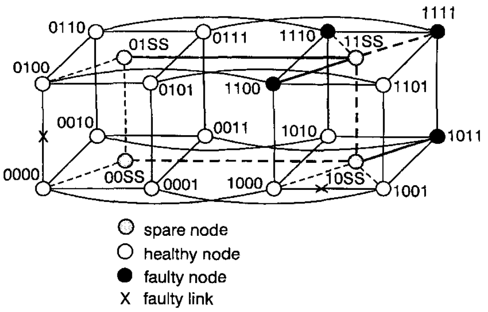

Fig. 3 *Reconfiguration of enhanced cluster hypercube*

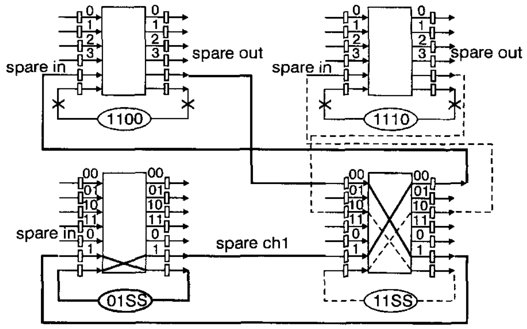

Fig. 4 *Replacing faulty nodes 1100 and 1110 with spare nodes 01SS and 11SS*

We assume that faulty nodes retain their ability to communicate. This assumption may be avoided by duplicating the communication module in each node. Moreover, we assume that the cost of communication is nearly constant between any two nodes with circuit-switched communication modules.

We next describe how the ECH tolerates faulty nodes and faulty links. Connecting a spare node to a regular node is done to tolerate a node failure. If the spare node resides in the cluster of the faulty node, the appropriate communication channel of the spare node is merged with the communication module of the faulty node. If the assigned spare node and the faulty node belong to different clusters, a dedicated path to connect them needs to be established. Once such a path is established due to the capabilities of the circuit-switched routing modules the physical location of the faulty node and its assigned spare node is irrelevant. Moreover, no modification of the available computation or communication algorithm is necessary. Faulty links are bypassed by establishing parallel paths using spare links. Fig. 3 illustrates the reconfiguration of an ECH with $d = 4$ and $i = 2$ in the presence of indicated faulty nodes and faulty links. By utilising the intermediate spare nodes, in effect, three logical spare nodes are present in cluster 11*XX*. Fig. 4 shows how spare nodes 01SS and 11SS replace faulty nodes 1100 and 1110, respectively, by merging their appropriate communication channels.

4 Theoretical results

Define a cluster with one or more faulty nodes as a faulty cluster. Since, within a cluster, the local spare node is directly connected to every regular node, the number of edge-disjoint paths between the faulty nodes of a cluster and the unassigned spare nodes in other clusters is the same as the number of edge-disjoint paths between the local spare node of the faulty cluster and the unassigned spare nodes. The reconfigurability of the ECH is then a function of the number of dedicated and edge-disjoint paths within the spare hypercube that can be established between the local spare node of a cluster with multiple faulty nodes and the available spare nodes in the fault-free clusters. We define the number of such edge-disjoint paths that must be constructed from a spare node as the *connection requirement* (C_R) of that spare node. For example, in Fig. 3, since two out of three logical spare nodes of cluster $11XX$ physically belong to other clusters, the C_R of the spare node $11SS$ is 2. The C_R of a spare node is equal to the number of faulty nodes in its cluster minus one.

The availability of these edge-disjoint paths is a connectivity issue of the spare hypercube. The following theorems examine the connectivity of the spare cube and therefore the reconfigurability of the ECH. We first set the upper bound on the number of faulty nodes that can be tolerated in a cluster. We then examine the lower bound on the number of faulty nodes that an ECH can tolerate for any fault distribution. Finally, we examine the conditions under which the maximum number of faulty nodes can be tolerated.

Theorem 1: The upper bound on the number of tolerated faulty nodes in a cluster that a d-dimensional ECH with cluster dimension of i can tolerate is $(d - i + 1)$.

Outline proof: Given a cluster with multiple faulty nodes, the local spare node can replace one of them. Since the local spare node has a degree of $(d - i)$ within the spare cube, at most $(d - i)$ edge-disjoint paths may be constructed from it to free spare nodes. ∎

Theorem 2: A d-dimensional ECH with cluster dimension of i can tolerate $(d - i + 1)$ faulty nodes regardless of the fault distribution.

Outline proof: The proof is by induction. $d = i$ represents the base case. The spare node can replace any faulty node directly. Therefore the ECH can tolerate $d - i + 1 = 1$ faulty node.

Next, consider an ECH of dimension $d = i + (k - 1)$. By the induction hypothesis, the $(i + k - 1)$-dimensional ECH can tolerate $(d - i) + 1 = (k - 1) + 1 = k$ faulty nodes. Consider an ECH of dimension $d = i + k$. Split the $(i + k)$-dimensional ECH along a dimension j such that $j > i$. Even if all $(k + 1)$ faulty nodes reside in the same $(i + k - 1)$-dimensional ECH, a dedicated path along dimension $(j - i)$ of the spare cube between the $(k + 1)$th faulty node and the unassigned spare node in the fault-free half of the ECH can be established. ∎

Group the spare nodes into three sets: S_S (set of source nodes), S_U (set of used nodes), and S_T (set of target nodes). A source node is a spare node in a cluster with multiple faulty nodes. The set S_S then represents the spare nodes with a C_R greater than 0. S_T is the set of unassigned spare nodes, and S_U consists of spare nodes that have been assigned to faulty nodes and have a C_R of 0. For example, considering only the faulty nodes in Fig. 3, after assigning the local spare node to a local faulty node in each faulty cluster, $S_S = \{11SS\}$, $S_U = \{10SS\}$, and $S_T = \{01SS, 00SS\}$. During the reconfiguration algorithm, discussed in Section 5, the spare nodes are dynamically assigned to the various sets. To illustrate this, suppose the C_R of a spare node $\alpha \in S_S$ is greater than 0 and there is a dedicated path from α to $\beta \in S_T$. Consequently, β replaces a faulty node in the cluster of α via the dedicated path. β is then called used and is assigned to S_U. Also, the C_R of α is reduced by one. If the C_R of α becomes zero, it is also marked as used and is assigned to S_U. The ECH is called reconfigured when S_S becomes an empty set.

The reconfigurability of the ECH is a function of the number of dedicated and edge-disjoint paths, within the spare cube, that can be established between the local spare nodes (nodes in S_S) of the clusters with multiple faulty nodes and the available spare nodes (nodes in S_T) of the fault-free clusters. However, spare nodes do not have to be interconnected as a cube. Obviously, if the spare nodes are interconnected as a complete graph, the ECH can tolerate up to $2^{(d-i)}$ regardless of fault distribution. Hence, the reconfigurability of the ECH is a direct consequence of the connectivity of the topology which interconnects the spare nodes. Represent the topology of the graph connecting the spare nodes by $G = (V, E)$ where $V = S_S \cup S_U \cup S_T$ and E consists of the appropriate spare links. Let the C_R of a node $n \in S_S$ be represented by $C_R(n)$, and denote the sum of the C_Rs of all nodes in a set P as $\Sigma_{n \in P} C_R(n)$. Since the number of faulty nodes cannot exceed the number of spare nodes $|S_T| \geq \Sigma_{n \in S_S} C_R(n)$. The following theorem examines the connectivity of G as it pertains to the reconfigurability of the ECH.

Theorem 3: Consider a graph G(V,E) where $V = S_S \cup S_U \cup S_T$. The necessary and sufficient condition for every node $n \in S_S$ to have $C_R(n)$ edge-disjoint paths to $C_R(n)$ nodes in S_T is that the minimum number of edges leaving any subset of nodes $P \subseteq V$ be greater than or equal to $\Sigma_{n \in (P \cap S_S)} C_R(n) - |P \cap S_T|$.

Outline proof: Consider a subset $P_1 \subseteq S_S$ (Fig. 5). Each of the edge-disjoint paths from a node in S_S to a node in S_T must be carried over at least one edge in the cutset $(P_1, V - P_1)$. Therefore sum of the C_Rs of the nodes in P_1 must be smaller than or equal to the number of edges in the cutset $(P_1, V - P_1)$. Next, consider a subset $P \subseteq V$ in which $|P \cap S_T|$ of the edge-disjoint paths may exist. The rest of the paths must then be carried over the cutset $(P, V - P)$. Therefore the necessary condition follows.

To prove the sufficient condition, create a new graph $G' = (V', E')$ by adding two nodes s and t to G as specified below and depicted by Fig. 6. Each node in S_T is connected to t via a single edge. Each node $n \in S_S$ is connected to s

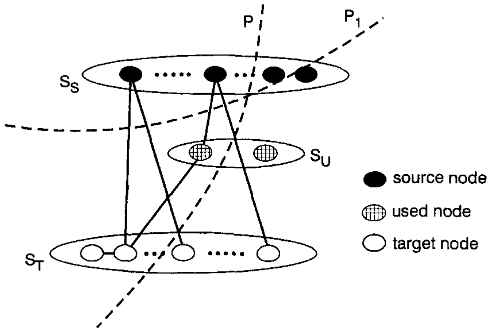

Fig. 5 *Cuts in graph G*

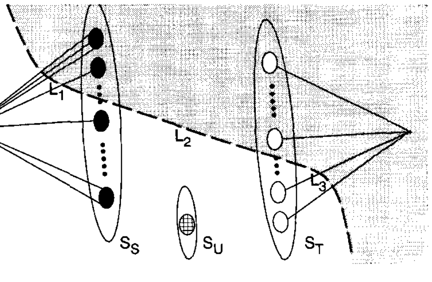

Fig. 6 *Cut in graph G'*

via $C_R(n)$ parallel edges. Let sum of the C_R of all nodes in S_S be L. Consider the depicted general cut in G' crossing indicated edges. Now examine the unshaded side of the cut. The given condition can then be formulated as $L_2 \geq (L - L_1) - L_3$ or $L_1 + L_2 + L_3 \geq L$. Hence, by Menger's theorem [14], L edge-disjoint paths exist between nodes s and t. ■

We next apply theorem 3 to the spare cube and examine the reconfigurability of the ECH. Theorems 4 and 5 set the bounds on the number of faulty nodes per cluster, under the maximum number of faulty nodes, that can be tolerated regardless of the fault distribution.

Theorem 4: Clusters with four or more faulty nodes can cause the reconfiguration of a d-dimensional ECH to fail.

Outline proof: Consider the case where every faulty cluster has four faulty nodes and faulty clusters form a subcube of dimension $(d-2)$. Subsequently, the spare nodes within the faulty clusters form a spare subcube of dimension $(d-i-2)$; the C_R of each spare node would be three. The number of spare links crossing the $(d-i-2)$ spare subcube is given as

$$((d-i)-(d-i-2)) * 2^{(d-i-2)} = 2 * 2^{(d-i-2)}$$

which is less than $3 \times 2^{(d-i-2)}$, as required by theorem 3.

Theorem 5: The ECH can tolerate $2^{(d-i)}$ faulty nodes with up to three faulty nodes per cluster regardless of the fault distribution.

Outline proof: The proof is similar to theorem 4. The worst-case scenario of every case is examined and is shown to satisfy theorem 3.

5 Experimental results

Some patterns of four faulty nodes per cluster can cause the reconfiguration of the ECH to fail. However, the probability that the faulty nodes can form such patterns is very low. Therefore a more realistic measure of the reconfigurability of the ECH would be under random fault distributions.

Next examine the reconfiguration algorithm. An optimal reconfiguration algorithm can be developed by utilising the maxflow/mincut algorithm [15]. Here, optimality is measured as the ability to assign a spare node to every faulty node whenever such an assignment is feasible *vis-a-vis* theorem 3. The main drawback to reconfiguration using the maxflow/mincut algorithm is that a new digraph has to be constructed and the spare node assignment has to be done by the host processor. To overcome these deficiencies, we next present a near optimal reconfiguration algorithm, which is called *Alloc-Spare.* The algorithm consists of three parts as follows:

(i) Early abort: The following solvability checks are performed to determine whether the reconfiguration is feasible. If the total number of faulty nodes is greater than the number of spare nodes $(2^{(d-i)})$, the reconfiguration fails. If the C_R of a spare node is greater than $(d-i)$, the reconfiguration fails due to theorem 1. The reconfiguration also fails if the sum of the C_R of any two neighbouring spare nodes in the spare cube is greater than $2(d-i)-2$ (theorem 3).

(ii) Local assignment: The local spare node of every faulty cluster is assigned to a faulty node within the cluster. If all faulty nodes are covered, the ECH is reconfigured.

(iii) Nonlocal assignment: To find a set of candidate spare nodes that can be assigned to a faulty node, we utilise Lee's path-finding algorithm [16]. The algorithm begins by constructing a breadth-first search of minimum depth j $(1 \leq j \leq 2^{(d-i)} - 1)$ in the spare cube from the local spare node of a faulty cluster with a non zero C_R. If a free spare node is found, a path to the spare is formed. The algorithm guarantees that a path to a spare node will be found if one exists and that the path will be the shortest possible [16]. Therefore all faulty nodes that are one link away from available spare nodes (at distance 1) are assigned first. Once a path is formed, the links associated with that path are deleted from the spare cube, resulting in a new structure. If there still remain some uncovered faulty nodes, a solvability test to check the C_R of neighbouring spare nodes, similar to Early Abort, is performed on the new structure, and step (iii) is repeated for a higher depth j. Reconfiguration fails if $j > (2^{(d-i)} - 1)$, which is the longest acyclic path in the spare cube. ■

We implemented algorithm Alloc-Spare for an ECH of dimension 20 with the spare cube of dimension 10. The simulation result for up to 1024 randomly placed faulty nodes is shown in Fig. 7 as plot G4. 1000 simulation runs were performed for each given number of faulty nodes. Plots G1, G2, and G3 in the Figure pertain to the schemes proposed in [4 and 17, 8, 7], respectively; compared with other schemes in the literature, the selected ones tolerate more faulty nodes for similar hardware overhead. The result indicates 100% reconfigurability for the ECH under the random fault distribution.

The required node degree of each regular and spare node in [17] is $(d+1)$ and 2^i respectively, which represents the minimal switch complexity among the presented schemes in Fig. 7. The next higher node degree belongs to the ECH with $(d+1)$ for the regular node and $2^i + (d-i)$ for the spare node. The regular and spare node degrees in both of the methods given in [7,8] have to be at least $(d+2)$ and $2^i + (d-i)$, respectively. Finally, the implementation of the scheme in [4] requires the highest switch complexity [7].

To examine the limitation of the ECH under random fault distribution, we next assumed that the number of faulty nodes in the ECH is the maximum $(2^{(d-i)})$. We then assumed that each faulty cluster contains a fixed number of faulty nodes. By theorem 1 a faulty cluster may have up to $(d-i+1)$ faulty nodes. Under the maximum number of faulty nodes, the number of faulty clusters is equal to the number of faulty nodes divided by the given number of faulty nodes per cluster. If the division results in a remainder, an additional cluster with the number of

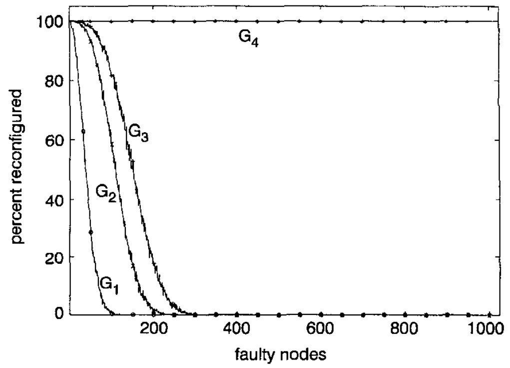

Fig. 7 *Reconfigurability of ECH under random fault distribution*

o G1 Cluster hypercube x G2 Alam & Melhem's chan.mux + G3 Alam & Melhem's sys2 *G4 Enhanced cluster hypercube. Dimension of hypercube = 20; dimension of each cluster = 10; spare nodes = 1024

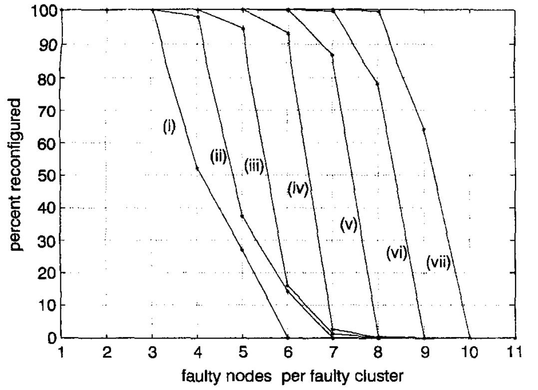

Fig. 8 *Reconfigurability of ECH under maximum number of faulty nodes*
(i) $D_S=4$ $N_f=16$ (ii) $D_S=5$ $N_f=32$ (iii) $D_S=6$ $N_f=64$ (iv) $D_S=7$ $N_f=128$ (v) $D_S=8$ $N_f=256$ (vi) $D_S=9$ $N_f=512$ (vii) $D_S=10$ $N_f=1024$

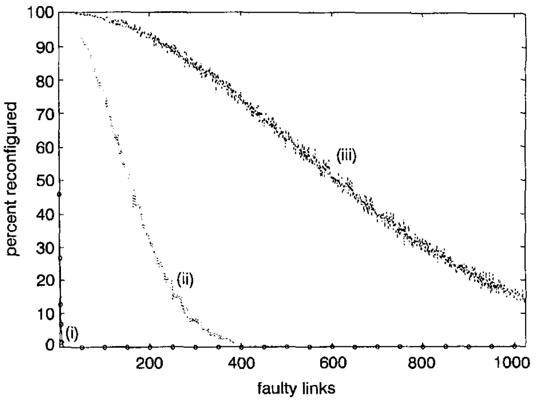

Fig. 10 *Reconfigurability under random fault distribution*
(i) cluster hypercube; (ii) ECH under MPP; (iii) ECH under dilation 2. Dimension of hypercube = 20; dimension of cluster = 10; spare nodes = 1024

faulty nodes equal to the remainder needs to be allocated, as well. The faulty clusters were then randomly allocated in the ECH, and the Alloc-Spare algorithm was applied to perform reconfiguration. The simulation results for an ECH of dimension 20 under different sizes of the spare cube are shown in Fig. 8, where each point in the graph represents the average of 1000 simulation runs. Fig. 8 indicates the percent number of cases where the ECH was reconfigured. The Figure shows that, under random fault distribution, an ECH with a large spare cube can nearly achieve 100% reconfiguration for up to $(d - i - 2)$ faulty nodes per cluster, which is much higher than the theoretical lower bound of three faulty nodes per cluster.

Next, we calculated the average number of spare links per spare node that is left unused after the reconfiguration. Our results, shown in Fig. 9, indicate that an ECH, under the maximum number of faulty nodes and the highest number of reconfigurable faulty nodes per cluster, uses on the average less than three spare links per spare node to reconfigure. Therefore the spare cube is a well-connected graph via the unused spare links even after the reconfiguration. For the case depicted in Fig. 7, the average number of utilised spare links per spare node is nearly 1.

As indicated in the previous Section, the ECH can also tolerate faulty links. However, no theoretical lower bound on the number of tolerated faulty links can be established, since two or more faulty links sharing a node in an ECH will cause the reconfiguration to fail. For example, in Fig. 3, if the links 0-00 and 000- are faulty, the reconfiguration fails, since the spare link 00*S*00 has to be used by two dedicated parallel paths to bypass the faulty links. Therefore we only examined experimental results based on random distribution of faulty links. We implemented two algorithms to tolerate faulty links. In the first one, called the minimum parallel path (*MPP*) algorithm, if any one of the spare links of a parallel path to any faulty link is unavailable, the reconfiguration fails. The second algorithm (dilation 2 algorithm) allows more than one parallel path to replace an intercluster faulty link. For example, in Fig. 3, if the spare link 0-*SS* is not available, the faulty link 0-00 can still be replaced by the parallel path 01*S*00→ − 1*SS*→1 − *SS*→ − 0*SS*→00*S*00. Fig. 10 shows the simulation results of our two algorithms and the scheme proposed in [17]. Other schemes [6,18] can only tolerate one faulty link. Fig. 10 suggests that about 100 faulty links can be tolerated by the MPP algorithm 90% of the time and nearly 100% of the time by the dilation 2 algorithm. The extra reconfigurability comes at the expense of slower reconfiguration.

By combining the reconfiguration algorithms which tolerate node failures and link failures, a combination of faulty links and faulty nodes is tolerated. For a faster reconfiguration we chose the MPP algorithm to tolerate faulty links. In our simulations we have assumed that the probability of having a faulty link is the same as having a faulty node.

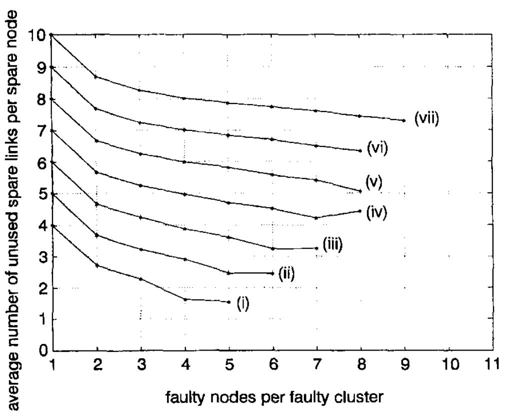

Fig. 9 *Average number of occured spare lenks per spare node*
(i) $D_S=4$ $N_f=16$ (ii) $D_S=5$ $N_f=32$ (iii) $D_S=6$ $N_f=64$ (iv) $D_S=7$ $N_f=128$ (v) $D_S=8$ $N_f=256$ (vi) $D_S=9$ $N_f=512$ (vii) $D_S=10$ $N_f=1024$

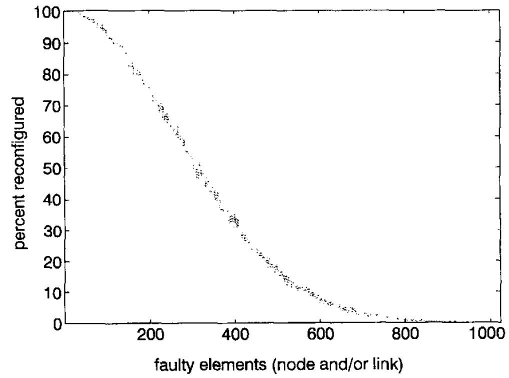

Fig. 11 *Reconfigurability under random fault distribution*
Dimension of hypercube = 20; dimension of cluster = 10; spare nodes = 1024

6 Conclusion

We have presented a strongly fault-tolerant design for a d-dimensional hypercube multiprocessor and examined its reconfigurability. Theoretical results indicate that our scheme can always tolerate the maximum number of faulty nodes with up to three faulty nodes per cluster. Our experimental results suggest that, under random fault distribution, the maximum number of faulty nodes can be tolerated with a very high probability. Since fast reconfiguration is very important in the presence of faults, our near-optimal reconfiguration algorithm is more appropriate than the optimal one, with negligible difference in the end result. Our scheme also tolerates a relatively large number of faulty links. Compared with other proposed schemes, the ECH can tolerate significantly more faulty nodes and faulty links for the same overhead.

8 References

1 TZENG, N.: 'A cube-connected cycles architecture with high reliability and improved performance,' *IEEE Trans.*, 1993, **C-42,** (2), pp. 246–253

2 DUTT, S., and J. P. HAYES: 'Designing fault-tolerant systems using automorphisms,' *J. Parallel Distrib. Comput.,* 1991, (12), pp. 249–268

3 BANERJEE, P., and PEERCY, M.: 'Design and evaluation of hardware strategies for reconfiguring hypercubes and meshes under faults,' *IEEE Trans.,* 1994, **C-43,** pp. 841–848

4 CHAU, C., and LIESTMAN, A. L.: 'A proposal for a fault-tolerant binary hypercubes architecture,' Proceedings of the IEEE international symposium on Fault-Tolerant Computing, Chicago, Illinois, USA, 1989, pp. 323–330

5 BANERJEE, P.: 'Strategies for reconfiguring hypercubes under faults,' Proceedings of the IEEE international symposium on Fault-Tolerant Computing, Newcastle-upon-Tyne, UK, 1990, pp. 210–217

6 BANERJEE, P., RAHMEH, J., STUNKEL, C., NAIR, V., ROY, K., BALASUBRAMANIAN, V., and ABRAHAM, J.: 'Algorithm-based fault tolerance on a hypercube multiprocessor,' *IEEE Trans.*, 1990, **C-39**, pp. 1132–1145

7 ALAM, M., and MELHEM, R.: 'An efficient modular spare allocation scheme and its application to fault tolerant binary hypercubes,' *IEEE Trans. Parallel Distrib. Syst.*, 1991, **2**, pp. 117–126

8 ALAM, M., and MELHEM, R.: 'Channel multiplexing in modular fault tolerant multiprocessors,' Proceedings of the IEEE international conference on Parallel Processing, Bloomington, Illinois, USA, 1991, pp. I492–I496

9 HAYES, J.P.: 'A graph model for fault-tolerant computing systems,' *IEEE Trans.,* 1976, **C-25,** pp. 875–884

10 BRUCK, J., CYPHER, R., and HO, C.T.: 'Efficient fault-tolerant mesh and hypercube architectures,' Proceedings of the 22nd annual international symposium on Fault-Tolerant Computing, Boston, Massachusetts, USA, July 1992, pp. 162–169

11 BRUCK, J., CYPHER, R., and HO, C.T.: 'Wildcard dimensions, coding theory and fault-tolerant meshes and hypercubes,' Proceedings of the 23nd annual international symposium on Fault-Tolerant Computing, Toulouse, France, July 1993, pp. 260–267

12 IZADI, B.: 'Design of fault-tolerant distributed memory multiprocessors' PhD thesis, 1995, Ohio State University

13 IZADI, B., and ÖZGÜNER, F.: 'A highly fault-tolerant circuit-switched hypercube,' Proceedings of IASTED international conference on Parallel and Distributed Systems, Barcelona, Spain, June 1997, pp. 9–14

14 COLBURN, C. J.: 'The combinatorics of network reliability' (Oxford University Press, 1987)

15 TUCKER, A.: 'Applied combinatorics' (Wiley, 1984, 2nd edn.)

16 LEE, C. Y.: 'An algorithm for path connection and its applications,' *IRE Trans.,* 1961, **EC-10,** pp. 346–365

17 IZADI B., and ÖZGÜNER, F.: 'Spare allocation and reconfiguration in a fault tolerant hypercube with direct connect capability,' Proceedings of the sixth conference on Distributed Memory Computing, Portland, Oregon, USA, April 1991, pp. 711–714

18 LATIFI, S.: 'Fault-tolerant hypercube multiprocessors,' *IEEE Trans. Reliab.*, 1990, **R-39** (3), pp. 361–368