A Real-Time Fault-Tolerant k-ary n-cube Multiprocessor

BABACK IZADI
Dept. of Elect. and Comp. Engineering
State University of New York
New Paltz, NY 12561 U.S.A.
bai@engr.newpaltz.edu
www.engr.newpaltz.edu/~bai

FÜSUN ÖZGÜNER
Dept. of Elect. Engineering
The Ohio State University
Columbus, Ohio 43210 U.S.A.
ozguner@ee.eng.ohio-state.edu
eewww.eng.ohio-state.edu/~ozguner

Abstract: - We present a real-time fault-tolerant design for the k-ary n-cube multiprocessor and examine its reconfigurability. The k-ary n-cube is augmented by spare nodes at stages one and two. We consider two modes of operations, one under heavy computation or hard deadline and the other under light computation or soft deadline. We assume that faulty nodes cannot compute, but retain their ability to communicate. Our approach utilizes the capabilities of the wave-switching communication modules of the spare nodes to tolerate a large number of faulty nodes and faulty links. Both theoretical and simulation results are examined. Compared with other proposed schemes, our approach can tolerate significantly more faulty nodes and faulty links with a low overhead and no performance degradation.

Key-Words: - Fault tolerance, real time, $k-$ary n-cube, spare allocation, reconfiguration, augmented multiprocessor, wave switching, circuit switching.

1 Introduction

In the quest to attain petascale computing, researchers are designing parallel machines with hundreds of thousands of processing elements [8]. The k-ary n-cube is an attractive topology to implement such parallel machines. A number of multiprocessors have already been built using networks that are either k-ary n-cube or are isomorphic to one [4, 15, 16]. To sustain the same level of performance, some researchers have investigated hardware schemes for the k-ary n-cube based multiprocessors where spare nodes and spare links are used to replace the faulty ones [3, 11, 12, 2, 1, 7, 5, 10, 17, 6]. To accommodate real-time applications, such faulty components have to be replaced with spares in a manner that also satisfies the required completion deadline of active tasks. Two modes of operation is generally considered: the *strict mode* and the *relaxed mode*. The strict mode pertains to tasks whose computational requirements are heavy or have a hard completion deadline. The relaxed mode, on the other hand, consists of tasks with a soft completion deadline or a light computational load. Therefore, in the strict mode of operation, in order to allow for fast reconfiguration, spare replacement of faulty components should result in very few changes in the system interconnections.A common approach to accommodate this mode of operation is to replace each faulty component with the local spare using a distributed reconfiguration algorithm [14]. On the other hand, in the relaxed mode of operation, a global reconfiguration algorithm is applied to maximize the probability that in the next strict mode of operation, there exists a local spare for every faulty component.

In this paper, we present a two-stage redundant scheme for the k-ary n-cube. The objectives of the scheme are two fold. First, facilitate real-time fault tolerance by allowing the system to operate in either the strict mode or the relaxed mode. Second, utilize the spare network to tolerate a large number of faculty nodes and faulty links.

The rest of the paper is organized as follows. In the next section, notation and definitions that are used throughout the paper are given. An overview of our approach is presented in Section 3. In Section 4, we examine the reconfigurability of the scheme. Both theoretical and simulation results are presented. Finally, concluding remarks are discussed in Section 5.

2 Notation and Definitions

Each node of a k-ary n-cube is identified by *n-tuple* $(a_{n-1} \cdots a_i \cdots a_0)$ where a_i is a radix k digit and represents the node's position in the i-th dimension. Each

node is connected along the dimension i to the neighboring nodes $(a_{n-1} \cdots (a_i \pm 1 \ mod \ k) \cdots a_0)$. Each spare node, in addition to n digits, at stage one is labeled with a prefix S (i.e. $Sa_{n-1} \cdots a_i \cdots a_0$) and at stage two is labeled with a prefix SS. The link connecting any two nodes P and Q is represented by $P \rightarrow Q$. A cluster whose local spare is labeled $Sa_{n-1} \cdots a_i \cdots a_0$ is called *cluster* $a_{n-1} \cdots a_i \cdots a_0$. Finally, we define the *Connection Requirement (C_R)* of a spare node in a cluster with multiple faulty nodes as the number of edge-disjoint paths that must be constructed, within the spare network from that spare node to other spare nodes in the fault-free clusters, so that faulty nodes can be tolerated.

3 Overview of the TECKN Approach

In our scheme, at stage one, one spare node is assigned to each set of i^n regular nodes called a cluster; each spare node is connected to every regular node of its cluster via an intra-cluster spare link. Furthermore, the spare nodes of neighboring clusters are interconnected using inter-cluster spare links; two clusters are declared neighbors if there exists at least one regular node in each cluster with a direct link between them. We call the resulting topology the *enhanced cluster k-ary n-cube* (*ECKN*) [11]. Hence, there exists $\frac{k^n}{i^n}$ spare nodes connected as a $\frac{k}{i}$-ary n-cube. Figure 1 depicts an ECKN with $k = 6$ and $n = i = 2$. At stage two,

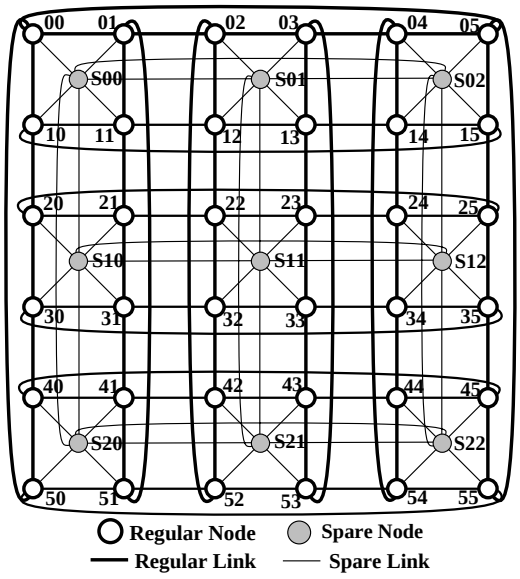

Figure 1: An enhanced cluster 6-ary 2-cube with $i = 2$

the process is repeated: the spare nodes of the $\frac{k}{i}$-ary n-cube at the first stage is also divided into j^n clusters and one spare node from the second stage is assigned to each cluster. Moreover, the spare nodes at the second stage are interconnected as a $\frac{k}{i \times j}$-ary n-cube. We call the resultant structure the *two-stage enhanced cluster k-ary n-cube* (*TECKN*). Figure 2 depicts a TECKN with $k = 8$ and $n = i = j = 2$. Hence, each regular node in the TECKN is connected to its $2n$ neighboring regular nodes and the local spare node at stage one. Each spare node at stage one is connected to i^n regular nodes of its local cluster, its assigned spare node at stage two, and its $2n$ neighboring spare nodes at stage one. Each spare node at stage two is connected to j^n spare nodes of its local cluster at stage one and its $2n$ neighboring spare nodes at stage two. Therefore, the degree of each regular node, each spare node at stage one, and each spare node at stage two are $(2n + 1)$, $(i^n + 2n + 1)$, and $(j^n + 2n)$, respectively.

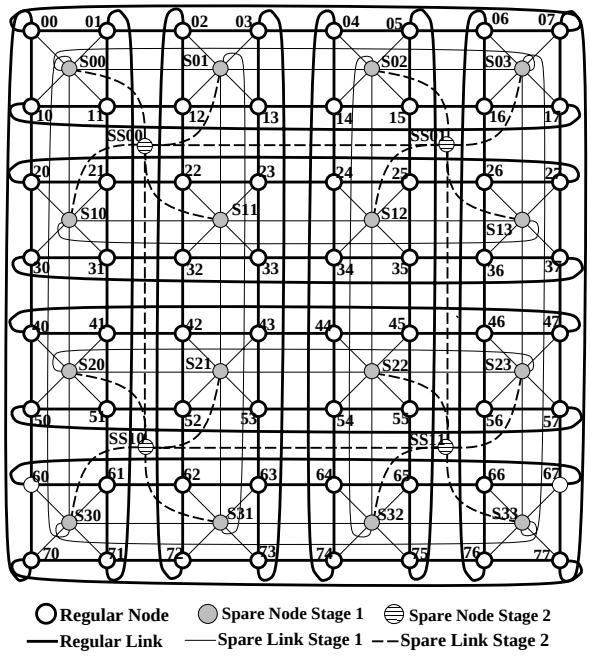

Figure 2: A two stage enhanced cluster 8-ary 2-cube with $i = j = 2$

We next describe how the TECKN tolerates faulty nodes and faulty links. Each node is made of a computation module and a wave-switching communication module [9]. Wave-switching implements circuit-switching and wormhole-switching concurrently; permanent connections and long messages use the circuit-switched segment while short messages are transmitted using the wormhole-switching. We assume that faulty nodes retain their ability to communicate. This is a common assumption since the hardware complexity of the communication module is much lower than the computational module. Therefore, the probability

of failure in the communication module is much lower than the computation module. This assumption may be avoided by duplicating the communication module in each node. To tolerate a faulty node, the computation module of the spare node logically replaces the computation module of the faulty node. In addition, if the spare node resides in the cluster of the faulty node, the new communication module consists of the functional communication module of the faulty node merged with the appropriate routing channel of the local spare node. If the assigned spare node and the faulty node belong to different clusters, a dedicated path is constructed by linking the appropriate routing channels of the intermediate spare nodes. Once such a path is established, due to the circuit-switched capability of the wave-switching communication modules, the physical location of the faulty node and its assigned spare node becomes irrelevant. Moreover, no modification of the available computation or communication algorithm is necessary. Similarly, faulty links are bypassed by establishing parallel paths using spare links. Figure 3 illustrates the reconfiguration of a TECKN with $k = 8$ and $n = i = j = 2$ in the presence of indicated faulty nodes and faulty links. For the sake of clarity, in Figure 3, non-active spare links are deleted and active spare links are drawn in a variety of line styles to distinguish the bypass paths. Note that by utilizing spare nodes from other fault-free clusters, in effect, four logical spare nodes are present in each of the clusters 10, 11, 12, and 31. Figure 4 illustrates how spare nodes S21 and S11 replace faulty nodes 32 and 22, respectively, by merging their communication modules. The light and dashed lines in Figure 4 pertain to similar lines in Figure 3 and represent effective permanent circuit-switched connections after the reconfiguration.

4 Reconfigurability of the TECKN

To allow for fast reconfiguration in the strict mode of operation, the reconfiguration algorithm should result in minimum changes in system interconnections. Hence, under the strict mode of operation, each faulty node of a cluster is replaced by the local spare node at stage one. The algorithm is applied distributively, allowing each spare node at stage one to monitor the status of the regular nodes within its cluster, and replace the faculty node as outlined in the previous section. The reconfiguration algorithm under the strict mode of operation fails if more than one node becomes faculty in a cluster.

The reconfiguration algorithm in the relaxed mode

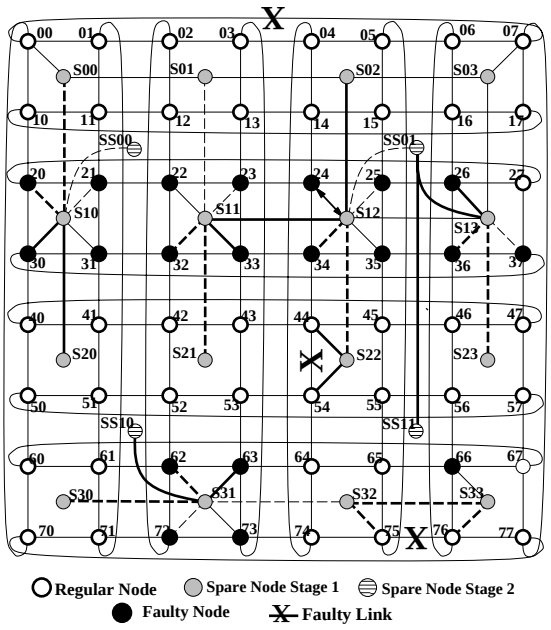

Figure 3: Reconfiguration of a TECKN

of operation first tries to assign each detected faulty node to a spare node at stage two. This is done to make the spare nodes at stage one available for the next strict mode of operation. For example, in Figure 2, in the relaxed mode of operation, the task of faulty nodes 22 and 62 are assigned to spare nodes $SS00$ and $SS10$ while in the strict mode of operation, they would be assigned to the local spare nodes $S11$ and $S31$, respectively. Under the relaxed mode of operation, if there is no available unassigned spare node at stage two, a spare node from stage one is assigned to the faulty node; details of reconfiguration algorithm under the relaxed mode of operation is discussed later in this section.

As indicated before, the reconfigurability of the TECKN is a function of the number of dedicated and edge-disjoint paths, within the spare network at stages one and two, 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. The availability of these edge-disjoint paths is a connectivity issue within the spare network. We next establish some bounds on the number of faulty nodes that a TECKN can tolerate.

Lemma 1 *A two-stage enhanced cluster k-ary n-cube ($k \neq 2$), can at most tolerate $2n + 2$ faulty nodes in one cluster.*

Proof: Given a cluster, at stage one, with multiple faulty nodes, the local spare node can replace one of

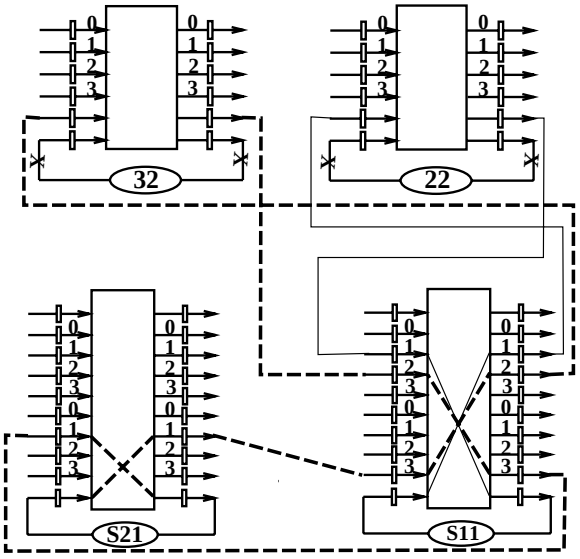

Figure 4: Replacing faulty nodes 22 and 32 with spare nodes S11 and S21, respectively

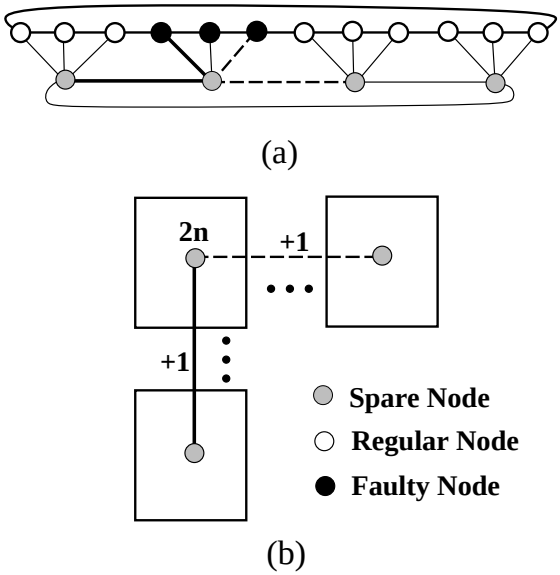

Figure 5: An enhanced cluster a) k-ary 1-cube b) k-ary $(n+1)$-cube

them. Since the local spare node, at stage one, has a degree of $(2n+1)$ within the spare network, at most $(2n+1)$ edge-disjoint paths may be constructed from it to unassigned spare nodes. ■

Theorem 1 *A two-stage enhanced cluster k-ary n-cube ($k \neq 2$), can tolerate $2n+2$ faulty nodes regardless of the fault distribution.*

Proof: We first show, by induction, that a TECKN at stage one can tolerate $(2n+1)$ faulty nodes regardless of the fault distribution. The TECKN at stage one is simply an ECKN. The base case is shown for $n = 1$. k regular nodes form a ring as depicted in Figure 5(a) for $k = 12$ and $i = 3$. Similarly, $\frac{k}{i}$ spare nodes form a ring. Hence, each spare node has a direct spare link connecting it to each of its two immediate neighboring spare nodes. By inspection, even if all $2 \times 1 + 1 = 3$ faulty nodes reside in the same cluster, there exists three edge-disjoint paths connecting the faulty nodes to the local spare node and its immediate neighboring spare nodes. Next, let us consider an ECKN of dimension $n+1$. By construction, an $(n+1)$-dimensional ECKN consists of k n-dimensional ECKN modules such that each spare node, in addition to its $2n$ spare links within its module, is connected to two spare nodes in other modules (Figure 5(b)). By the induction hypothesis, each ECKN of dimension n can tolerate $2n+1$ faulty nodes. Suppose there exist $2(n+1)+1 = 2n+3$ faulty nodes. If the distribution of faulty nodes is such that at most $2n+1$ faulty nodes reside in one ECKN of dimension n, the system can tolerate them by the induction hypothesis. Consider the case where all faulty nodes reside in the same n-dimensional ECKN module. $2n+1$ of the faulty nodes can be tolerated locally by the induction hypothesis. Furthermore, since every faulty node has an unused spare link to its local spare node and the local spare node has two unused spare links connecting it to two free spare nodes in other modules (unassigned spare nodes within two fault-free ECKN of dimension n), two dedicated and edge-disjoint paths between the last two faulty nodes and the unassigned spare nodes can be established. The system can therefore tolerate $2(n+1)+1$ faulty nodes and it follows by induction that $(2n+1)$ faulty nodes can be tolerated within the spare network at stage one.

Finally, consider the TECKN at stages one and two. Since the $(2n+2)$th faulty node has an available edge connecting it to the local spare node, at stage one, and the local spare node has an unused spare link connecting it to a spare node at stage 2, a dedicated path between the $(2n+2)$th faulty node and the spare node at stage 2 can be established. The system can therefore tolerate $(2n+2)$ faulty nodes regardless of the distribution of faulty nodes. ■

Since there is no bound on radix or dimension of the TECKN, no higher theoretical bound on the number of tolerated faulty nodes can be established [11]. To have a better realistic measure of fault-tolerant capability of the TECKN, we simulated the reconfigurability of the TECKN, under random fault distribution, based on the

following reconfiguration algorithm. An optimal reconfiguration algorithm can be developed by utilizing the maxflow algorithm [18]. The main drawback to a reconfiguration using the above algorithm is that a digraph representation of the spare network has to be constructed [11] 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. The algorithm consists of four parts as specified below:

1. 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 ($(\frac{k}{i})^n + (\frac{k}{i\times j})^n$), the reconfiguration fails. If the C_R of a spare node at stage one is greater than $(2n+1)$, the reconfiguration also fails due to Lemma 1.

2. Assignment at stage two: We utilize Lee's path-finding algorithm [13] to find a set of candidate spare nodes at stage two that can be assigned to the faulty node. The algorithm begins by constructing a breadth-first search of minimum depth d ($1 \leq d \leq ((\frac{k}{i\times j})^n)$) in the spare network of stage two from the local spare node of a faulty cluster. If a free spare node is found, a path is formed to the faulty node. The algorithm guarantees that a path to a spare node will be found if one exits and the path will be the shortest possible [13]. Once a path is formed, the links associated with that path are deleted from the spare tree, resulting in a new structure. If there still remain some uncovered faulty nodes, a solvability test similar to step 1 is performed on the new structure and this step is repeated for a higher depth d in stage two of the spare cube.

3. Local assignment: If all spare nodes at stage two are assigned and there still remain some faulty nodes, the local spare node of every faulty cluster is assigned to a faulty node within the cluster.

4. Assignment at stage one: If there remain additional faulty nodes, we apply Lee's path-finding algorithm to both stages one and two from the local spare node of a faulty cluster with an unassigned faulty node. Reconfiguration fails if $d > (\frac{k}{i\times j})^n + (\frac{k}{i})^n$, which is the longest acyclic path in the spare network. ■

We implemented the reconfiguration algorithm for a TECKN with $k = 66$, $i = 11$, $j = 2$, and $n = 3$. 1000 simulation runs were performed for each given number of faulty nodes. The result of our simulations, under the random fault distribution, indicate 100% reconfiguration in the presence of up to 243 faulty nodes (the maximum). Similar results were attained under different cluster sizes at stages one and two.

We next examined the limitation on the reconfigurability of the TECKN under random fault distribution as well as the effect of the second stage (and perhaps additional stages) on the fault-tolerant capability of the TECKN. We thus assumed $(\frac{k}{i\times j})^n + (\frac{k}{i})^n$ faulty nodes in the TECKN and $((\frac{k}{j})^n)$ faulty nodes in the TECKN with only the first stage of spare network (ECKN). Moreover, we assumed that each faulty cluster contains a fixed number of faulty nodes. Figure 6 depicts the result of our simulation for a 66-ary 3-cube with the radix of each cluster at stage one and stage two being eleven and two, respectively. The solid and the dashed lines in the figure pertain to the result of the TECKN under 243 faulty nodes and the ECKN under 216 faulty nodes, respectively. The result indicates that, under the maximum number of faulty nodes, the TECKN can tolerate up to six faulty nodes per cluster 100% of the time. In addition, it demonstrates that the TECKN can handle one more faulty node per cluster than the ECKN. The result is intriguing since the degree of the spare node of the TECKN at stage one is also higher than the ECKN by one. Similar results were attained under different dimension sizes and cluster sizes at stages one and two. Our simulation results, therefore, illustrate that the existence of a second stage of redundancy is more critical to the higher reconfigurability of the TECKN than the size of the spare network at the second stage. Hence, multi-stage redundancy should only marginally enhance the reconfigurability of the k-ary n-cube over the TECKN, since the degree of the spare node at stage one is $(2n+1)$ regardless of the number of spare stages or their dimensions.

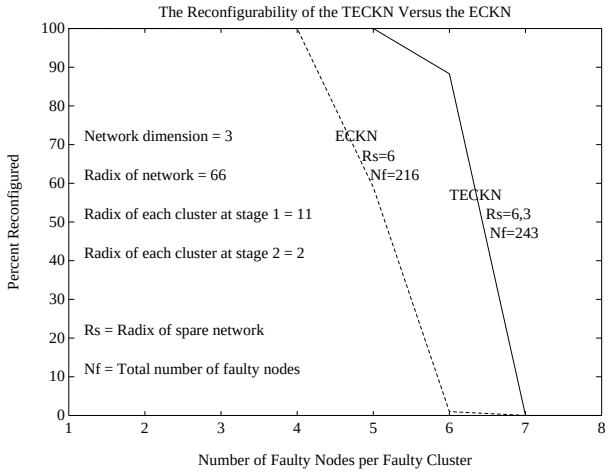

Figure 6: Reconfigurability of the TECKN versus the ECKN

As indicated in the previous section, the TECKN can also tolerate faulty links. However, some faulty links can only have one bypass path. Therefore, no distinction between their relaxed and strict mode of operations can be made. Also, two or more faulty links sharing a node in a TECKN will cause the reconfiguration to fail. Hence, no theoretical lower bound on the number of tolerated faulty links can be established. Our simulation results of the TECKN, based on random distribution of faulty links, is slightly better than the simulation results of the ECKN [11].

5 Conclusion

In this paper, we proposed a practical scheme to allow a k-ary n-cube multiprocessor tolerate faulty nodes in real time. During the strict mode of operation, the scheme uses local reconfiguration, which is the fastest and involves the fewest switch changes. Then, in the next relaxed mode of operation the tasks of local spare nodes, at stage one, are transferred to the spare nodes at the second stage by applying a global reconfiguration scheme. If a node becomes faulty during the relaxed mode of operation, the scheme tries to assign a spare node at stage two to replace it. This is done to maximize the probability that in the next strict mode of operation local spare nodes may be available to replace the potential faulty nodes. In the relaxed mode of operation, if there is no available spare node at stage two, spare nodes of stage one are utilized to replace the faulty nodes. Our theoretical results indicate that, in the relaxed mode of operation, our scheme can always tolerate the $2n + 2$ faulty nodes regardless of their distribution. Our experimental results suggest that, under random fault distribution, $(\frac{k}{i \times j})^n + (\frac{k}{i})^n$ faulty nodes (the maximum) can be tolerated with a very high probability. In the strict mode of operation, one faulty node per cluster can be tolerated. Hence, a smaller cluster size at stage one is needed if the strict mode has a harder deadline and vice versa; other factors such as the switch complexity of the routers also influence the size of the clusters at stages one and two.

References

[1] M. Alam and R. Melhem. Channel multiplexing in modular fault tolerant multiprocessors. *Proceedings of the IEEE International Conference on Parallel Processing*, pp. I492–I496, 1991.

[2] M. Alam and R. Melhem. Routing in modular fault-tolerant multiprocessor systems. *IEEE Transactions on Parallel and Distributed Systems*, 6:1206–1220, November 1995.

[3] M. Bae and B. Bose. Spare processor allocation for fault tolerance in torus-based multicomputers. *Proceedings of the Twenty-sixth International Symposium on Fault-Tolerant Computing*, pp. 282–290, 1996.

[4] J. Brandenburg. Technology advances in the Intel Paragon system. *Annual ACM Symposium on Parallel Algorithms and Architectures*, pp. 182–182, June 1993.

[5] J. Bruck, R. Cypher, and C. T. Ho. Efficient fault-tolerant mesh and hypercube architectures. *Proceedings of the 22nd Annual International Symposium on Fault Tolerant Computing*, pp. 162–169, July 1992.

[6] J. Bruck, R. Cypher, and C. T. Ho. Fault-tolerant meshes with small degree. *Proceeding of 5th ACM Symposium on Parallel Algorithm and Architectures*, pp. 1–10, June 1993.

[7] S. Chakravarty and S. J. Upadhyaya. A unified approach to designing fault-tolerant processor ensembles. *Proceedings of the IEEE International Conference on Parallel Processing*, pp. 339–342, 1988.

[8] J. Dongarra and D. Walker. The quest for petascale computing. *IEEE Computing in Science and Engineering*, pp. 32–39, May 2001.

[9] J. Duato, P. Lopez, and S. Yalamanchili. Deadlock- and livelock-free routing protocols for wave switching. In *Proceedings of the 11th International Parallel Processing Symposium*, pp. 570–577, April 1997.

[10] S. Dutt. Fast polylog-time reconfiguration of structurally fault-tolerant multiprocessors. *IEEE Symposium on Parallel and Distributed Processing*, pp. 161–169, 1993.

[11] B. Izadi and F. Özgüner. Design of a circuit-switched highly fault-tolerant k-ary n-cube. *Proceedings of the 1997 International Conference on Parallel Processing*, pp. 342–345, August 1997.

[12] H. Ku and J. Hayes. Systematic design of fault-tolerant multiprocessors with shared buses. *IEEE Transactions on Computers*, 46(14):439–455, April 1997.

[13] C. Y. Lee. An algorithm for path connection and its applications. *IRE Transactions on Electronic Computers*, ec-10:346–365, 1961.

[14] R. Libeskind-Hadas, N. Shrivastava, R. Melhem, and C. Liu. Optimal reconfiguration algorithms for real-time fault-tolerant processor arrays. *IEEE Transactions on Parallel and Distributed Systems*, 6:498–510, May 1995.

[15] M. Noakes, D. Wallach, and W. Dally. The J-machine multicomputer: An architectural evaluation. *Proceedings of the IEEE International Symposium on Computer Architecture*, pp. 224–235, May 1993.

[16] S. Scott and G. Thorson. The Cray T3E network: Adaptive routing in a high performance 3d torus. *proceedings of HOT Interconnects IV*, August 1996.

[17] N. Tsuda. Fault-tolerant processor arrays using additional bypass linking allocated by graph-node coloring. *IEEE Transactions on Computers*, 49(5):431–442, May 2000.

[18] A. Tucker. *Applied Combinatorics 2nd ed.* Wiley, 1984.