JOURNAL OF PARALLEL AND DISTRIBUTED COMPUTING **40,** 185–193 (1997)
ARTICLE NO. PC961279

Minimum Assignment of Test Links for Hypercubes with Lower Fault Bounds[1]

DAJIN WANG*,[2] AND ZHONGXIAN WANG†

Department of Mathematics and Computer Science and †Department of Information and Decision Sciences, Montclair State University, Upper Montclair, New Jersey 07043

In an n-dimensional hypercube multiprocessor system, to correctly diagnose faulty processors among themselves, the maximum allowed number of faulty processors is n under the well-known PMC diagnostic model. When the n fault bound is adopted, all links between processors will be used in the diagnosis. However, if the fault bound is lower than n, many links can be freed from the task of performing diagnosis. In this paper, we show that each drop of the fault bound by 1 will free 2^{n-1} links from diagnosis. We will present an algorithm that selects, in a symmetric manner, the to-be-freed links, so that only a minimum number of links will be used to perform diagnosis. A rigorous proof for the algorithm's correctness is given. The freed links will never be used for the purpose of diagnosis, so that the diagnosis and some conventional computations may be carried out simultaneously, improving the performance of the system as a whole.

1. INTRODUCTION

The hypercube structure is a well-known interconnection model. As a topology to interconnect a multiprocessor system, it has been proved to possess many attractive properties. Multiprocessor computers built with hypercube structure have been already in existence [6, 9, 10, 15]. Because of its importance for achieving high performance, the fault-tolerant computing for hypercube structures has been the interest of many researchers. The recent work includes [3, 4, 8, 12, 14], etc. Generally speaking, the fault tolerance is achieved either by providing spare processors or by computing in the presence of faulty processors. No matter which strategy of the two is used, as the first step to deal with faults, the system has to discriminate the faulty processors from fault-free ones. The process of determining faulty processors is called the *diagnosis* of the system. Since the processors in a system detect among themselves, clearly the processors cannot all be faulty at the time of diagnosis. So all diagnosis strategies assume an *a priori fault bound*, which is the maximum allowed number of faults at a time.

It is a well-known fact that the diagnosability of an n-dimensional hypercube (n-cube for short) under PMC model (see definition in the next section) is n; i.e., an n-cube can correctly detect all faulty nodes, provided that the number of faulty nodes does not exceed n. When the adopted fault bound is maximum, n, all the links will be involved in the diagnosis. As the technology progresses rapidly, the failure probability of each processor steadily drops. Consequently, the real fault bound may be well lower than n. When the fault bound is less than n, it is no longer necessary to use all links when the system performs diagnosis. This gives the motivation to use less links for the purpose of diagnosis, so that the diagnosis and some regular computations may be carried out concurrently. A processor can switch between testing mode (using the links designated for diagnosis) and computing mode (using links not participating in diagnosis). It is then a natural question to ask which links are designated for diagnosis while the remaining ones are for nondiagnosis computation. In this paper we show that for an n-cube whose fault bound is known to be $n - k$, $1 \leq k \leq n - 2$, only $(n - k)2^{n-1}$ links are needed for the testing among nodes. An efficient algorithm is presented to symmetrically select the minimally needed $(n - k)2^{n-1}$ test links for an $(n - k)$-fault-bounded n-cube. A rigorous proof is given to show that the algorithm works correctly. It is shown that the $(n - k)2^{n-1}$ links chosen by the algorithm always induce a symmetric-structured, $(n - k)$-connected subgraph of the n-cube, thus satisfying the sufficient condition for a system to be diagnosable provided that the fault bound is $n - k$.

The rest of this paper is organized as follows. Section 2 gives the necessary backgrounds and defines the terminology used in the paper. Section 3 presents the algorithm that chooses the $(n - k)2^{n-1}$ test links that form a connected subgraph of n-cube that is $(n - k)$-diagnosable. Also in Section 3, the correctness of the algorithm is proved. We give some concluding remarks in Section 4.

2. PRELIMINARIES

An n-dimensional hypercube, denoted Q_n, is an undirected graph $G(V, E)$ such that V consists of 2^n nodes, numbered from $\underbrace{00...0}_{n}$ to $\underbrace{11...1}_{n}$, and an edge (or link)

[1] The algorithm given in this paper was presented, without correctness proof, at the 14*th IEEE International Phoenix Conference on Computers and Communications*, Phoenix, Arizona, March 28–31, 1995.

[2] E-mail: wang@pegasus.montclair.edu.

0743-7315/97 $25.00

$\{v_i, v_j\} \in E$ iff v_i and v_j have exactly one bit different. Thus, each node has immediate links with exactly n other nodes, and it is easy to establish that $|E| = n2^{n-1}$.

An n-dimensional hypercube, or n-cube for short, can be divided into several *subcubes*. Formally, an m-dimensional subcube can be defined as $s_n s_{n-1} \ldots s_1$, where exactly m s_i's are x, and the rest $(n - m)$ s_i's are either 0 or 1. A node with number $b_n b_{n-1} \ldots b_1$ in $s_n s_{n-1} \ldots s_1$ is such that

- $s_i = 0 \Rightarrow b_i = 0$,
- $s_i = 1 \Rightarrow b_i = 1$,
- $s_i = x \Rightarrow b_i = 0$ or $b_i = 1$.

An m-dimensional subcube, thus, consists of 2^m nodes. For example, $x1x0$ is the subcube containing nodes {0100, 0110, 1100, 1110} and all links among these nodes. Two subcubes are said to be *disjoint* if they have no nodes in common. By the definition of n-cube, the two nodes linked by an edge have one and only one bit different. So this edge can be uniquely represented using the two nodes it links. If $v_i = b_n..b_k..b_1$, $v_j = b_n..\overline{b}_k..b_1$, then we denote edge $\{v_i, v_j\}$ as $b_n..b_{k+1}Xb_{k-1}..b_1$. We call $b_n..b_{k+1}Xb_{k-1}..b_1$ an edge of dimension k. There are 2^{n-1} edges in each dimension.

There are several strategies for interconnected processors to diagnose faulty processors at system level among themselves. In this paper, we are concerned with one strategy, called *one-step diagnosis*, that was initially proposed by Preparata *et al.* [13]. In the diagnostic model introduced in [13] (known as the PMC model), the self-diagnosable system is represented by a directed graph $G = (V, A)$, or digraph for short, with nodes representing processors and arrows representing the tests performed among processors. Node v_i can test all nodes v_j if arrow $(v_i \rightarrow v_j) \in A$. An undirected graph $G = (V, E)$ is just a special case of a digraph $G = (V, A)$ in which $(v_i \rightarrow v_j) \in A \Leftrightarrow (v_j \rightarrow v_i) \in A$. The test-result, regardless of the detailed testing technique, is simply a conclusion that the tested node is "faulty" or "fault-free," denoted as label 1 or 0 on the corresponding arrow. The PMC model assumes that a fault-free node should always give a correct test-result, whereas, the test-result given by a faulty node is unreliable. A *syndrome* is defined as a function $s : A \rightarrow \{0, 1\}$. A subset $F \subseteq V$ is *consistent* with a syndrome s if s can arise from the circumstance that all nodes in F are faulty and all nodes in $V - F$ are fault-free. It is worth pointing out that for a given syndrome s, there may be *more than one* subset of V that are consistent with s. If this happens, the system can not diagnose for syndrome s, because the faulty sets that can cause s are not unique. It is clear that for the PMC model (and also for many other models), we must have some (at least one) good processors to self-diagnose.

For the PMC diagnostic model, [13] proposed a one-step diagnosis strategy. Its target is to identify, before any replacement of faulty nodes, a unique set of nodes F such that all nodes in F are faulty and all nodes in $V - F$ are fault-free. Notice that under this strategy, the diagnosis is carried out among nodes of V with faulty nodes present; i.e., it does not matter whether the tester is faulty or fault-free—every tester just performs tests on its testees, and the overall result will be analyzed to reach a conclusion about which nodes are "really faulty" and which are "really fault-free." One-step diagnosis algorithms under PMC have been proposed by many researchers since the establishment of the model, which can be found in [1]. Obviously, for this diagnosis strategy, to have a reliable diagnostic result, the number of faulty nodes cannot exceed a certain limit. The *diagnosability* under a certain strategy is a maximum integer t such that when the number of faulty processors is less than or equal to t, the diagnosis can be carried out successfully. More formally,

DEFINITION 1. Under the PMC model, a system is said to be (one-step) t_p-diagnosable if for any syndrome s, there is at most one faulty-subset $F \subseteq V$ that is consistent with s, given that the number of faulty nodes does not exceed t_p.

Clearly the diagnosability is very much dependent on the topology of the testing assignment, i.e., the underlying digraph.

In a hypercube-structured system, two linked processors can directly access each other and therefore can perform a test on each other. If $G(V, E)$ represents the structure of the hypercube system and $\{v_i, v_j\} \in E$, we call v_i a tester of v_j, and v_j a tester of v_i. The testing assignment is therefore the same as the topology of the system structure. The following theorem gives the diagnosability of an n-cube.

THEOREM 1. [1, 11]. *A system of n-cube-structure is n-diagnosable.*

We also need the following definition for the discussion in this paper.

DEFINITION 2. The connectivity $\kappa(G)$ of a graph $G(V, E)$ is the minimum number of nodes whose removal results in a disconnected or a trivial (one node) graph.

3. TEST-DELETION ALGORITHM FOR HYPERCUBES OF LOWER FAULT BOUNDS

The n-diagnosability of hypercube in Theorem 1 was derived based on two earlier results, which gave the necessary and sufficient conditions of a t_p-diagnosable system, respectively. Two conditions are necessary for a system S of N processors to be t_p-diagnosable [13]:

1. $N \geq 2t_p + 1$,
2. Each processor is tested by at least t_p other processors.

Two sufficient conditions for S to be t_p-diagnosable are [7]:

1. $N \geq 2t_p + 1$
2. $\kappa(G) \geq t_p$,

where G is the undirected graph representing S's interconnection, and $\kappa(G)$ is the connectivity of G. It was shown in [1] that $\kappa(G) = n$ if G is an n-cube, and therefore by the above necessary and sufficient conditions, the n-cube is n-diagnosable.

From the necessary conditions, for an n-cube to be n-diagnosable, every node should be tested by at least n other nodes. On the other hand, every node in an n-cube has links to just n other nodes. That is to say, for an n-cube to be n-diagnosable, all $n2^{n-1}$ links will be involved in the diagnosis. $n2^{n-1}$ is the minimum number of links to have n-diagnosability.

Since an n-cube is n-diagnosable, as many as n nodes can be allowed faulty for the system to self-diagnose correctly. The fault bound of an n-cube is therefore n. As the technology advances rapidly, the failure probability of each processor drops considerably. For an n-cube, it may well be that the fault bound is less than n. When the fault bound is lower, it is no longer necessary to have all links participate in the diagnosis. For instance, if the fault bound is decreased by 1 (to $n - 1$), then as many as 2^{n-1} links can be freed from the task of performing diagnosis. More generally, we have

LEMMA 1. *If the fault bound of an n-cube is n − k, then the necessary number of links for the diagnosis is* $(n - k)2^{n-1}$.

Proof. The necessary conditions for a system to be $(n - k)$-diagnosable require that each node be tested by at least $n - k$ other nodes. It is equivalent to saying that every node should be at least of degree $n - k$. If the least possible degree $n - k$ is adopted at every node, then since there are 2^n nodes in an n-cube, $\sum_{n \in V} degree(v) = 2^n(n - k)$. By a fundamental graph theory theorem ("The First Theorem of Graph Theory"), for an undirected graph $G(V, E)$, $\sum_{v \in V} degree(v) = 2|E|$ [5]. Now let E_k be the edge-set of an $(n - k)$-diagnosable n-cube that perform diagnosis. We have $2|E_k| = 2^n(n - k)$, giving $|E_k| = (n - k)2^{n-1}$. ■

Lemma 1 tells us that only $(n - k)2^{n-1}$ links are necessary for having $(n - k)$-diagnosability. This suggests that there may be a way to choose just that many links (from the total $n2^{n-1}$ links) for performing fault diagnosis. In the rest of this paper, we will present a simple systematic algorithm that chooses $(n - k)2^{n-1}$ links in such a way that not only every node is of degree $n - k$ (necessary condition for $(n - k)$-diagnosability), but also the subgraph induced from the $(n - k)2^{n-1}$ links is $(n - k)$-connected (sufficient condition for $(n - k)$-diagnosability).

For our purpose, we list all $n2^{n-1}$ links of an n-cube in n columns, with column i containing all links of dimension i. The 1st column lists the 2^{n-1} links in the following "increasing" order:

$$\overbrace{000...00X}^{n}$$
$$000...01X$$
$$......$$
$$111...11X.$$

The ith column, $2 \le i \le n$, is obtained by left-rotating one bit for all links of $(i - 1)$th column. For example, the listing of all edges of a 5-cube is:

column 1	*column* 2	*column* 3	*column* 4	*column* 5
0000X	000X0	00X00	0X000	X0000
0001X	001X0	01X00	1X000	X0001
0010X	010X0	10X00	0X001	X0010
0011X	011X0	11X00	1X001	X0011
0100X	100X0	00X01	0X010	X0100
0101X	101X0	01X01	1X010	X0101
0110X	110X0	10X01	0X011	X0110
0111X	111X0	11X01	1X011	X0111
1000X	000X1	00X10	0X100	X1000
1001X	001X1	01X10	1X100	X1001
1010X	010X1	10X10	0X101	X1010
1011X	011X1	11X10	1X101	X1011
1100X	100X1	00X11	0X110	X1100
1101X	101X1	01X11	1X110	X1101
1110X	110X1	10X11	0X111	X1110
1111X	111X1	11X11	1X111	X1111

When the fault bound is n, all edges are needed for diagnosis. When the fault bound drops by 1 (to $n - 1$), 2^{n-1} can be discriminated from others so that they will not be used in diagnosis. If the fault bound drops further by 1, another 2^{n-1} can be discriminated, etc. The following algorithm symmetrically specifies the $k2^{n-1}$ links to be removed when the fault bound is $n - k$. After the removal, the remaining $(n - k)2^{n-1}$ links are minimally necessary for future diagnosis tasks.

3.1. Algorithm Description

The algorithm takes as input the complete edge set, listed in n columns as shown above. The 2^{n-1} edges in a column are referred as 1st edge, 2nd edge, ..., in top-down order.

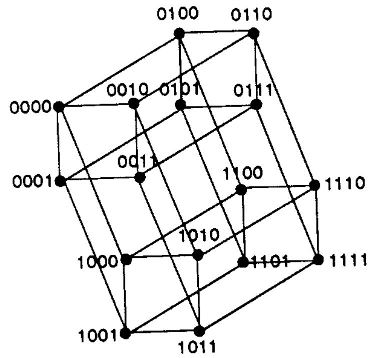
ALGORITHM REMOVAL.

{Purpose: Remove $k2^{n-1}$ links for an n-cube with $n - k$ fault bound}

Input: $n - k$ ($k \le n - 2$) the fault bound; the complete edge-set of n-cube
Output: the remaining $(n - k)2^{n-1}$ edges

for $j = 1$ **to** k **do**
  1. at column j,
      for $i = 1$ **to** 2^{n-j-1} **do**
          remove the ith edge
      od
  2.1. at column $j + 1$,
      for $i = 2^{n-j-2} + 1$ **to** 2^{n-2} **do**
          remove the ith edge
      od
  2.2 at column $j + 1$,
      for $i = 2^{n-2} + 1 + 2^{n-j-2}$ **to** 2^{n-1} **do**
          remove the ith edge
      od
od

Notice that the algorithm runs for all fault bound ≥ 2.



column 1	column 2	column 3	column 4
$000X$	$00X0$	$0X00$	$X000$
$001X$	$01X0$	$1X00$	$X001$
$010X$	$10X0$	$0X01$	$X010$
$011X$	$11X0$	$1X01$	$X011$
$100X$	$00X1$	$0X10$	$X100$
$101X$	$01X1$	$1X10$	$X101$
$110X$	$10X1$	$0X11$	$X110$
$111X$	$11X1$	$1X11$	$X111$

FIG. 1. The 4-cube and its edge set.

As an example, we apply the algorithm to a 4-cube. The 4-cube and its complete edge set are show in Fig. 1. If $k = 1$, after *Removal*, the remaining edge set and the corresponding incomplete 4-cube are shown in Fig. 2. One can check by inspection that it is a 3-connected graph. When $k = 2$, another 2^{n-1} edges are removed by *Removal*, resulting in the edge set and its corresponding incomplete 4-cube shown in Fig. 3. It can be observed that the remaining edges form a ring that traverses all nodes, and the graph is 2-connected. In extreme cases, when the fault bound drops to 1, we can simply remove any one edge from the ring and the resulting graph is 1-connected and therefore 1-diagnosable.

It is also interesting to notice the structure of the removed $k2^{n-1}$ links, since they can be used in nondiagnostic, regular computing. We have $k2^{n-1} = 2^{n-k} \times k2^{k-1}$, where $k2^{k-1}$ is the number of links of a k-cube. It is observed that when $k = 1$, the 2^{n-1} removed links make up 2^{n-1} disjoint 1-cubes. When $k \geq 2$, the $2^{n-k} \times k2^{k-1}$ links cannot induce 2^{n-k} disjoint k-cubes. However, they can induce 2^{n-k-1} disjoint k-cubes in subcube $1xxx..xx$. Since it is quite often that an application will use only some subcubes of a hypercube machine, the diagnosis and some regular computations may be carried out concurrently, with processors being programmed to switch between testing mode (using the links designated for diagnosis) and computing mode (using links not participating in diagnosis).

3.2. The Correctness Proof of Algorithm

To prove that algorithm *Removal* works correctly, we resort to the sufficient conditions for a system to be $(n-k)$-diagnosable. In other words, we will prove that the remaining edges make up a graph whose connectivity is $n - k$. We need Whitney's theorem for that purpose.

THEOREM 2. [16]. *A graph has connectivity m if and only if there exist at least m disjoint paths between every pair of nodes in the graph.*

LEMMA 2. *The algorithm* **Removal** *breaks all direct links between the following 2^k $(n-k-1)$-cubes in subcube $0xxx..xx$*

$0xx..x\,\underbrace{x00000..00}_{k+1}$
$0xx..x\,\underbrace{x10000..00}_{k+1}$ } 2 $(n-k-1)$-*cubes*

$0xx..x\,\underbrace{0x1000..00}_{k+1}$
$0xx..x\,\underbrace{1x1000..00}_{k+1}$ } 2

$0xx..x\,\underbrace{00x100..00}_{k+1}$
......
$0xx..x\,\underbrace{11x100..00}_{k+1}$ } 4

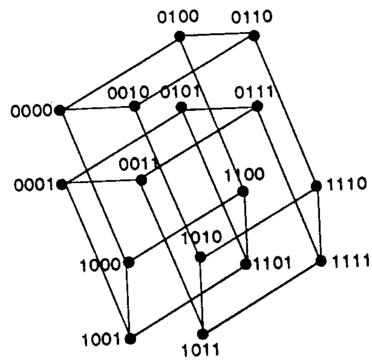
$0xx..x\,\underbrace{000x10..00}_{k+1}$
......
$0xx..x\,\underbrace{111x10..00}_{k+1}$ } 8

⋮

$0xx..x\,\underbrace{000..000x1}_{k+1}$
......
$0xx..x\,\underbrace{111..111x1}_{k+1}$ } 2^{k-1}

and breaks all direct links between the following 2^k $(n-k-1)$-cubes in subcube $1xxx..xx$

$1xx..x\underbrace{00..0}_{k}x$
$1xx..x\underbrace{00..1}_{k}x$
......
$1xx..x\underbrace{11..1}_{k}x$ } 2^k



column 1	*column* 2	*column* 3	*column* 4
	$00X0$	$0X00$	$X000$
	$01X0$	$1X00$	$X001$
		$0X01$	$X010$
		$1X01$	$X011$
$100X$	$00X1$	$0X10$	$X100$
$101X$	$01X1$	$1X10$	$X101$
$110X$		$0X11$	$X110$
$111X$		$1X11$	$X111$

FIG. 2. The 4-cube after running *Removal* for $k = 1$. 2^{n-1} tests (edges) have been removed. The graph is 3-connected.

The disjointness of the subcubes in $1xxx..xx$ is obvious. To see that the subcubes in $0xxx..xx$ are disjoint, only notice that for any two subcubes, there is at least one non x bit that is different. We now prove the lemma by induction on k.

Proof. When $k = 1$, at 1st column, *Removal* deletes the first 2^{n-2} edges of dimension 1. These are *all* edges of dimension 1 in subcube $0xxx..xx$. So subcube $0xxx..xx$ is disconnected into subcubes $0xxx..x0$ and $0xxx..x1$. At 2nd column, *Removal* removes the second and the fourth 2^{n-3} edges of dimension 2. From the listing of edges we know that these are *all* edges of dimension 2 in subcube $1xxx..xx$. The deletion therefore disconnected $1xxx..xx$ into $1xxx.x0x$ and $1xxx..x1x$. The basis case has been checked out.

HYPOTHESIS. *The claim of the lemma holds for k.*

INDUCTION. *Consider $k + 1$. When $j = k + 1$, the step* 1 *of in the algorithm becomes*:

> at column $k + 1$,
> **for** $i = 1$ **to** 2^{n-k-2} **do**
> remove the ith edge;

i.e., the first 2^{n-k-2} links at column $k + 1$ (dimension $k + 1$) are removed. By hypothesis, $0xx..x\underbrace{x00000..00}_{k+1}$ is an $(n - k - 1)$-subcube that has no connection to any other subcubes in $0xxx..xx$. Each dimension of the subcube has 2^{n-k-2} links. Observe that the first 2^{n-k-2} links at column $k + 1$ are $0\underline{0..00}\underbrace{X0..00}_{k+1}$ through $0\underline{1..11}\underbrace{X0..00}_{k+1}$, which are just *all* links of dimension $k + 1$ in $0xx..x\underbrace{x00000..00}_{k+1}$. So the removal of those links disconnects it into two smaller subcubes

$$0xx..x\underbrace{x000000..00}_{k+2}$$

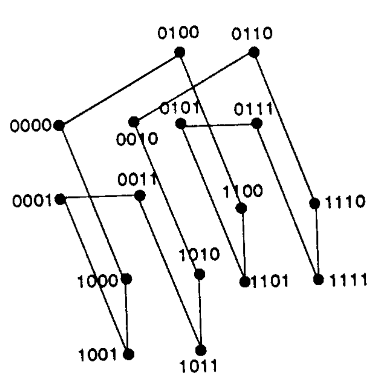
$$0xx..x\underbrace{x100000..00}_{k+2}.$$

Step 2 of the algorithm becomes

> 2.1. at column $k + 2$,
> **for** $i = 2^{n-k-3} + 1$ **to** 2^{n-2} **do**
> remove the ith edge;
> 2.2. at column $k + 2$,
> **for** $i = 2^{n-2} + 1 + 2^{n-k-3}$ **to** 2^{n-1} **do**
> remove the ith edge;

At column $k + 2$, the first 2^{n-k-3} links are

$$0\underline{0..00}\underbrace{X0..00}_{k+2},\ ...,\ 0\underline{1..11}\underbrace{X0..00}_{k+2}. \qquad (1)$$



column 1	*column* 2	*column* 3	*column* 4
		$0X00$	$X000$
			$X001$
			$X010$
			$X011$
$100X$	$00X1$	$0X10$	$X100$
$101X$	$01X1$		$X101$
$110X$			$X110$
$111X$			$X111$

FIG. 3. The 4-cube after running *Removal* for $k = 2$. $2 \cdot 2^{n-1}$ tests (edges) have been removed. The graph is 2-connected.

The 2^{n-k-3} links starting from $2^{n-2}+1$ are

$$0\underline{0..00}\underbrace{X1..00}_{k+2},\ ...,\ 0\underline{1..11}\underbrace{X1..00}_{k+2}. \tag{2}$$

The algorithm removes all links of dimension $k+2$ *except* the $2 \cdot 2^{n-k-3}$ links in (1) and (2). Removal of links

$$0\underline{0..00}\underbrace{Xb_m..b_1 0\overbrace{10..0}^{d}}_{k+2},\ ...,\ 0\underline{1..11}\underbrace{Xb_m..b_1 0\overbrace{10..0}^{d}}_{k+2}.$$

and

$$0\underline{0..00}\underbrace{Xb_m..b_1 1\overbrace{10..0}^{d}}_{k+2},\ ...,\ 0\underline{1..11}\underbrace{Xb_m..b_1 1\overbrace{10..0}^{d}}_{k+2}.$$

will disconnect subcube $0xx..x\underbrace{b_m..b_1x\overbrace{100..00}^{d}}_{k+1}$ into two smaller subcubes

$$0xx..x\underbrace{0b_m..b_1x\overbrace{100..00}^{d}}_{k+2}$$

$$0xx..x\underbrace{1b_m..b_1x\overbrace{100..00}^{d}}_{k+2}$$

Since d ranges from 1 to k, *every* subcube in $0xxx..xx$ as listed in the lemma is disconnected into two smaller subcubes. The induction for subcube $0xxx..xx$ is thus complete.

As for link removal in subcube $1xxx..xx$, notice that the algorithm removes *all* 2^{n-2} links of dimension $k+2$ in $1xxx..xx$. Removal of links

$$1\underline{0..00}\underbrace{X0..0}_{k+1}0,\ ...,\ 1\underline{1..11}\underbrace{X0..0}_{k+1}0.$$

and links

$$1\underline{0..00}\underbrace{X0..0}_{k+1}1,\ ...,\ 1\underline{1..11}\underbrace{X0..0}_{k+1}1$$

will disconnect subcube $1xx..x\underbrace{00..0}_{k}x$ into two smaller subcubes

$$1xx..x\underbrace{000..00}_{k+1}x$$

$$1xx..x\underbrace{100..00}_{k+1}x.$$

Similarly, removal of links

$$1\underline{0..00}\underbrace{X0..01}_{k+1}0,\ ...,\ 1\underline{1..11}\underbrace{X0..01}_{k+1}0$$

and links

$$1\underline{0..00}\underbrace{X0..01}_{k+1}1,\ ...,\ 1\underline{1..11}\underbrace{X0..01}_{k+1}1$$

will disconnect subcube $1xx..x\underbrace{00..1}_{k}x$ into two smaller subcubes

$$1xx..x\underbrace{000..01}_{k+1}x$$

$$1xx..x\underbrace{100..01}_{k+1}x.$$

The arguments go as above and finally we get the subcube $1xx..x\underbrace{11..1}_{k}x$ disconnected into

$$1xx..x\underbrace{011..11}_{k+1}x$$

$$1xx..x\underbrace{111..11}_{k+1}x$$

The whole induction step is thus complete. ■

By Lemma 2, after link deletion by *Removal*, in both subcubes $0xxx..xx$ and $1xxx..xx$, there are 2^k $(n-k-1)$-cubes without any links among them. To go from one subcube to another, we must use links of dimension n, i.e., those linking $0xxx..xx$ and $1xxx..xx$.

LEMMA 3. *After the link-deletion by Removal, each subcube in* $0xxx..xx$ *has* 2^{n-k-1} *links of dimension n to exactly two subcubes in* $1xxx..xx$ (2^{n-k-2} *links to each subcube*). *Similarly, each subcube in* $1xxx..xx$ *has* 2^{n-k-1} *links of dimension n to exactly two subcubes in* $0xxx..xx$.

Proof. We prove the first assertion first. Arbitrarily pick a subcube $0xx..x\underbrace{b_m..b_1x10..00}_{k+1}$ from $0xxx..xx$. (Every subcube is of this form except the first one.) It has 2^{n-k-2} links to subcube $1xx..x\underbrace{b_m..b_1010..0x}_{k+1}$: The links are $X\underline{00..0}b_m..b_1010..00$ through $X\underline{11..1}b_m..b_1010..00$. It has 2^{n-k-2} links to subcube $1xx..x\underbrace{b_m..b_1110..0x}_{k+1}$: The links are $X\underline{00..0}b_m..b_1110..00$ through $X\underline{11..1}b_m..b_1110..00$. For subcube $0xx..x\underbrace{x00..000}_{k+1}$, the subcubes it has links to are $1xx..x\underbrace{000..00x}_{k+1}$ and $1xx..x\underbrace{100..00x}_{k+1}$, respectively.

We now prove the second assertion. Pick subcube $1xx..x\underbrace{b_kb_{k-1}..b_2b_1x}_{k+1}$. It has 2^{n-k-2} links to $0xx..x\underbrace{b_kb_{k-1}..b_2x1}_{k+1}$.

The links are $X\underline{00..0}b_k..b_2b_11$ through $X\underline{11..1}b_k..b_2b_11$. To specify the other subcube it has links to, we classify two cases. *Case* 1: $b_kb_{k-1}..b_2b_1 = 00..00$. Then the subcube is $0xx..x\underbrace{x000..00}_{k+1}$. The links are $X\underline{00..0}\underbrace{000..00}_{k+1}$ through $X\underline{11..1}\underbrace{000..00}_{k+1}$. *Case* 2: $b_kb_{k-1}..b_2b_1 > 00..00$. Let l, $k \geq l \geq 1$, be the smallest number such that $b_l = 1$, i.e., $b_kb_{k-1}..b_2b_1 = b_k..b_{l+1}10..0$. Then the subcube is $0xx..x\underbrace{b_k..b_{l+2}x10..0}_{k}0$. The links are $X\underline{00..0}\underbrace{b_k..b_{l+1}10..0}_{k}0$ through $X\underline{11..1}\underbrace{b_k..b_{l+1}10..0}_{k}0$. ■

The relationship between subcubes and links of dimension n is illustrated in Fig. 4. It can be seen that a ring of subcubes is formed, with every consecutive two linked by a group of 2^{n-k-2} links.

LEMMA 4. *Given a complete n-cube. For any node v_0, there exist n disjoint paths from v_0 to any other n nodes v_i, $i = 1, 2, ..., n$.*

Proof. The lemma is proved by induction on n. For small n (e.g., $n = 2$ or 3), the lemma can be easily verified through inspection.

HYPOTHESIS. *For an $(n-1)$-cube, there exist $n-1$ disjoint paths from v_0 to any other $n-1$ nodes v_i, $i = 1, 2, ..., n-1$.*

INDUCTION. *Given an n-cube Q_n. Consider the two subcubes* 0xx..x *and* 1xx..x *of which Q_n is composed. Without loss of generality, let v_0 belong to* 0xx..x *and number (label) $v_0 = 000..0$. Let $V' = \{v_1, v_2, ..., v_n\} \subset V$ be an arbitrary node-set such that $v_0 \notin V'$.*

Case 1. *All nodes of V' fall in* 0xx..x. For a subset of V', $\{v_1, v_2, ..., v_{n-1}\}$, by hypothesis, there are $n-1$ disjoint paths from v_0 to $v_1, v_2, ..., v_{n-1}$, using only nodes in $0xx..x$. We can always assume that v_n is not on any of these paths. (If v_n happens to be on the path from v_0 to, say, v_i, then we can name v_i to be v_n, and vice versa.) Let v_n's numbering (labeling) be $v_n = 0b_{n-1}..b_1$. Using only intermediate nodes

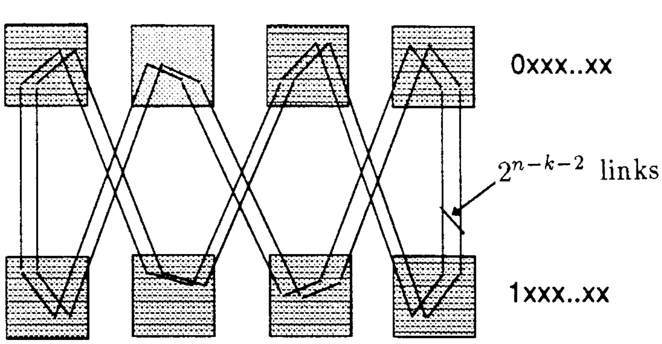


FIG. 4. The relationship between subcubes and links of dimension n.

in subcube $1xx..x$, we can construct the required path as follows:

$$v_0 = 000..0 \to 100..0 \to ... \to 1b_{n-1}..b_1 \to 0b_{n-1}..b_1 = v_n.$$

Case 2. *All nodes of V' fall in* 1xx..x. By hypothesis, there are $n-1$ disjoint paths from 100..0 to $\{v_1, ...v_{n-1}\}$, only using nodes of subcube $1xx..x$. Denote the numbering of the $n-1$ nodes adjacent to 100..0 on these paths $1\tilde{b}_1$, ..., $1\tilde{b}_{n-1}$, respectively, where $\tilde{b}_1$, etc., represent the remaining $n-1$ bits of a node. Denote the numbering of $v_n = 1\tilde{b}_n$. v_n does not lie on any of the $n-1$ paths. (If v_n lies on the path from 100..0 to v_i, just rename v_n to be v_i, and vice versa.) By hypothesis again, there are $n-1$ disjoint paths from v_0 to $0\tilde{b}_2$, ..., $0\tilde{b}_{n-1}$, $0\tilde{b}_n$. The n disjoint paths from v_0 to $\{v_1, ...v_{n-1}, v_n\}$ can then be constructed as

$$v_0 \to 100..0 \rightsquigarrow v_1$$
$$v_0 \rightsquigarrow 0\tilde{b}_2 \to 1\tilde{b}_2 \rightsquigarrow v_2$$
$$......$$
$$v_0 \rightsquigarrow 0\tilde{b}_{n-1} \to 1\tilde{b}_{n-1} \rightsquigarrow v_{n-1}$$
$$v_0 \rightsquigarrow 0\tilde{b}_n \to 1\tilde{b}_n = v_n.$$

Case 3. $\{v_1, ...v_k\}$ *fall in* 0xx..x. $\{v_{k+1}, ...v_n\}$ *fall in* 1xx..x, *where* $1 \leq k < n$. Denote the n target nodes in two subcubes as

In subcube $0xx..x$: $0\tilde{a}_1, ..., 0\tilde{a}_x, 0\tilde{b}_1, ..., 0\tilde{b}_y$

In subcube $1xx..x$: $1\tilde{b}_1, ..., 1\tilde{b}_y, 1\tilde{c}_1, ..., 1\tilde{c}_z$,

where $x + y = k$ and $y + z = n - k$. Node $1\tilde{b}_1$ has $n-1$ neighbors (i.e., directly linked nodes) in $1xx..x$. At least $(n-1) - [x + (y-1) + z] = y$ of them do not intersect with $\{1\tilde{a}_1, ..., 1\tilde{a}_x, 1\tilde{b}_2, ..., 1\tilde{b}_y, 1\tilde{c}_1, ..., \tilde{c}_z\}$. We choose one and denote it $1\tilde{b}'_1$. Similarly, for $1\tilde{b}_2$, there will be at least y such neighbors. Since one of them may have been chosen by $1\tilde{b}_1$, there are $y-1$ for $1\tilde{b}_2$ to choose. In summary, for nodes $1\tilde{b}_i$, $i = 1, ..., y$, we can choose a unique neighbor $1\tilde{b}'_i$ such that it does not fall into $\{1\tilde{a}_1, ..., 1\tilde{a}_x, 1\tilde{b}_1, ..., 1\tilde{b}_y, 1\tilde{c}_1, ..., 1\tilde{c}_z\}$. It is equivalent to saying that we can find a subset $\{0\tilde{b}'_1, ..., 0\tilde{b}'_y\}$ in $0xx..x$ such that $\{0\tilde{b}'_1, ..., 0\tilde{b}'_y\} \cap \{0\tilde{a}_1, ..., 0\tilde{a}_x, 0\tilde{b}_1, ..., 0\tilde{b}_y, 0\tilde{c}_1, ..., 0\tilde{c}_z\} = \phi$.

Case 3.1. $100..0 \notin \{v_{k+1}, ...v_n\}$. By hypothesis, there are $n-1$ disjoint paths from v_0 to $0\tilde{b}'_2, ..., 0\tilde{b}'_y$, $0\tilde{a}_1$, ..., $0\tilde{a}_x$, $0\tilde{b}_1$, ..., $0\tilde{b}_y$, $0\tilde{c}_1$, ..., $0\tilde{c}_z$, using only nodes in $0xx..x$. We can then construct $n-1$ disjoint paths from $v_0 - 1$ targets as

$$v_0 \rightsquigarrow 0\tilde{a}_i, i = 1, ..., x$$
$$v_0 \rightsquigarrow 0\tilde{b}_i, i = 1, ..., y$$
$$v_0 \rightsquigarrow 0\tilde{b}'_i \to 1\tilde{b}'_i \to 1\tilde{b}'_i, i = 2, ..., y$$
$$v_0 \rightsquigarrow 0\tilde{c}_i \to 1\tilde{c}_i, i = 1, ..., z.$$

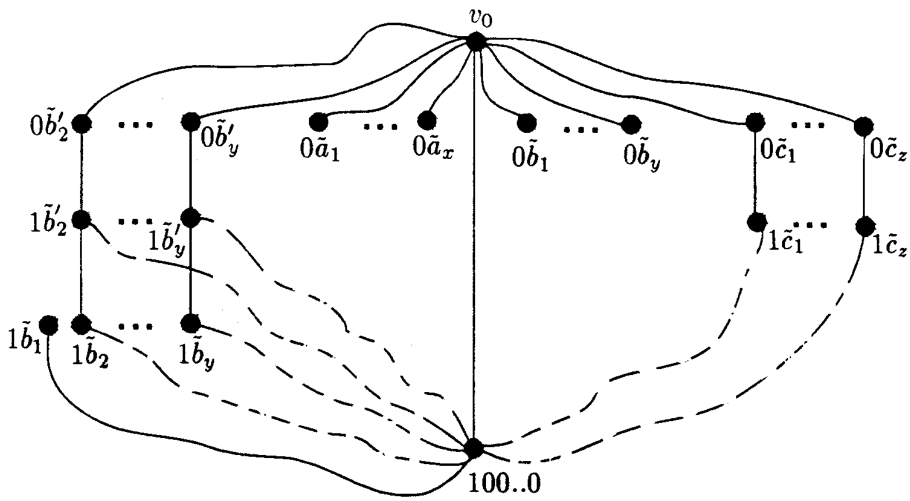


FIG. 5. The illustration of Case 3.1: n disjoint paths from v_0 to $0\tilde{a}_1$, .., $0\tilde{a}_x$, $0\tilde{b}_1$, .., $0\tilde{b}_y$, $1\tilde{b}_1$, .., $1\tilde{b}_y$, $1\tilde{c}_1$, .., $1\tilde{c}_z$.

For the path to $1\tilde{b}_1$, consider the $2y + z - 1$ nodes $1\tilde{b}_i$, $i = 1, ..., y$, $1\tilde{b}'_i$, $i = 2, ..., y$, $1\tilde{c}_i$, $i = 1, ..., z$. Since $2y + z - 1 = n - x - 1 \leq n - 1$, by hypothesis, there are $n - x - 1$ disjoint paths from 100..0 to them, using only nodes of $1xx..x$. So there is a path

$$v_0 \to 100..0 \rightsquigarrow 1\tilde{b}_1$$

which does not contain any nodes used in other $n - 1$ paths. See Fig. 5 for the illustration.

Case 3.2. $100..0 \in \{v_{k+1}, ...v_n\}$. Let $1\tilde{c}_1 = 100..0$. By hypothesis, there are $n - 1$ disjoint paths from v_0 to $0\tilde{b}'_1, ..., 0\tilde{b}'_y, 0\tilde{a}_1, ..., 0\tilde{a}_x, 0\tilde{b}_1, ..., 0\tilde{b}_y, 0\tilde{c}_2, ..., 1\tilde{c}_z$, using only nodes in $0xx..x$. We can construct n disjoint paths from v_0 to n targets as

$$v_0 \rightsquigarrow 0\tilde{a}_i, i = 1, ..., x$$

$$v_0 \rightsquigarrow 0\tilde{b}_i, i = 1, ..., y$$

$$v_0 \rightsquigarrow 0\tilde{b}'_i \to 1\tilde{b}'_i \to 1\tilde{b}_i, i = 1, ..., y$$

$$v_0 \rightsquigarrow 0\tilde{c}_i \to 1\tilde{c}_i, i = 2, ..., z$$

$$v_0 \rightsquigarrow 100..0 = 1\tilde{c}_1.$$

This completes the induction step, thus completing the proof of the lemma. ■

The following Corollary immediately follows Lemma 4.

COROLLARY 1. *Given a k-subcube Q_k in an n-cube. For any node v_0 in Q_k, there exist k disjoint paths, using only links in Q_k, from v_0 to any other k nodes in Q_k.*

LEMMA 5. *After the link-deletion by Removal, the remaining incomplete hypercube is $(n - k)$-connected.*

Proof. We will show that for any two nodes in the incomplete hypercube, there exist $n - k$ disjoint paths. Then by Theorem 2 it is $(n - k)$-connected.

Let v_0, v_1 be two arbitrary nodes. By Lemma 2, the algorithm *Removal* disconnects the original n-cube into 2^{k+1} $(n - k - 1)$-cubes, 2^k in $0xxx.xx$, 2^k in $1xxx..xx$. To traverse among these subcubes, one has to use links of dimension n.

Case 1. *v_0 and v_1 are in the same $(n - k - 1)$-cube, say Q_x.* Since an $(n - k - 1)$-cube is $(n - k - 1)$-connected, there exist $(n - k - 1)$ disjoint paths form v_0 to v_1 such that the paths only use edges in Q_x. For the one more path, refer to Fig. 4. By Lemma 3, the 2^{k+1} $(n - k - 1)$-cubes form a ring (of subcubes), linked by 2^{n-k-2} edges of dimension n between two cubes, and with alternative ones in $0xxx..xx$ and $1xxx..xx$. It is then obvious that there exists one path from v_0 to v_1 using edges out of Q_x.

Case 2. *v_0 in Q_x, v_1 in Q_y, $x \neq y$.* Let the subcube to which v_0 has an n-dimensional link be Q_z. Pick $n - k - 1$ nodes in Q_x, $v_x^1, ..., v_x^{n-k-1}$, such that

- the n-dimensional edges connected to them go to the same subcube $Q_{z'}$ and
- $Q_{z'} \neq Q_z$.

By Corollary 1, there exist $n - k - 1$ disjoint paths from v_0 to $v_x^1, ..., v_x^{n-k-1}$, using only links in Q_x. v_x^i $(i = 1, ..., n - k - 1)$ traverse through $Q_{z'}$, and then the next subcube in the subcube ring, ..., until they reach Q_y. (Notice that during the traversal, the $n - k - 1$ paths are all "parallel," without using any edges in common.) Let the $n - k - 1$ "arrival" nodes in Q_y be $v_y^1, ..., v_y^{n-k-1}$. Again by Corollary 1, there exist $n - k - 1$ disjoint paths from v_1 to $v_y^1, ..., v_y^{n-k-1}$, using only links in Q_y. The $n - k - 1$ disjoint paths from v_0 to v_1 are thus established.

For the one more path, we take the opposite direction in the ring: v_0 goes to a node in Q_z through an n-dimensional link. The path then traverses all subcubes until it reaches a node v'_0 in $Q_{y'}$, the subcube before Q_y. v_1 has an n-dimensional link to a node v'_1 in $Q_{y'}$. Obviously there is a path from v'_0 to v'_1, using only edges in $Q_{y'}$. Since this

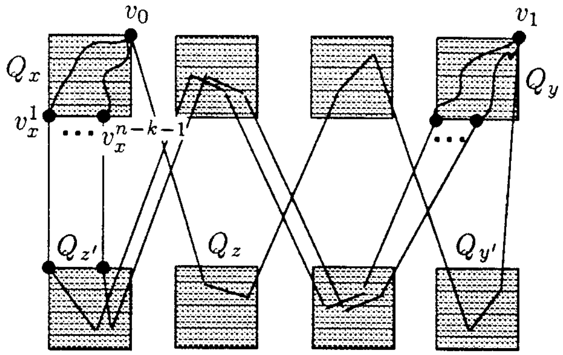


FIG. 6. The $n - k$ disjoint paths from v_0 to v_1.

path takes the ring direction opposite to the previous $n - k - 1$ disjoint paths, it uses no edges common to them. The situation is illustrated in Fig. 6. ■

THEOREM 3. *After the link-deletion by Removal, the remaining incomplete hypercube is* $(n - k)$*-diagnosable.*

Proof. We show that after the link deletion by *Removal*, the sufficient conditions for a system to be $(n - k)$-diagnosable are met.

Condition 1. $N \geq 2(n - k) + 1$, where N is the number of nodes in the system. This condition is trivially met in a hypercube.

Condition 2. $\kappa(G) \geq (n - k)$, where G is the system's underlying graph. By Lemma 5, this condition is met. ■

4. CONCLUDING REMARKS

We have presented an algorithm that specifies, in an n-cube, the edges that can be freed from taking part in diagnosis when the adopted fault bound is not maximum. The algorithm symmetrically removes $k2^{n-1}$ edges so that the 2^n nodes with the remaining $(n - k)2^{n-1}$ edges make up an $(n - k)$-connected graph. The validity of the algorithm was proved. It was shown that the $(n - k)2^{n-1}$ links chosen by the algorithm always induce an $(n - k)$-connected subgraph of the n-cube, thus satisfying the sufficient condition for a system to be diagnosable provided that the fault bound is $n - k$. We point out that the $(n - k)$-connectivity is a sufficient condition for $(n - k)$-diagnosability. (A *disconnected*, $(n - k)$-diagnosable subgraph could be obtained by simply removing *all* links of dimensions 1 through k). So this algorithm gives not only an $(n - k)$-diagnosable system, but at the same time an induced subgraph that is symmetric-structured and $(n - k)$-connected, which is a property of interest theoretically and practically as well.

Received June 19, 1995; accepted April 12, 1996

REFERENCES

1. Armstrong, J. R., and Gray, F. G. Fault diagnosis in a Boolean n cube array of microprocessors. *IEEE Trans. Comput. C* **30**(8) (1981), 587–590.
2. Bruck, J., Cypher, R., and Ho, C.-T. Efficient fault-tolerant mesh and hypercube architectures. *Proc. 22nd Int. Symp. Fault-Tolerant Comput.* July 1992, pp. 162–169.
3. Bruck, J., Cypher, R., and Soroker, D. Running algorithms efficiently on faulty hypercubes. *Comput. Arch. News* **19**(1)(Mar. 1991), 89–96.
4. Chartrand, G., and Lesniak, L. *Graphs and Digraphs.* Wadsworth, Belmont, CA, 1986.
5. Hillis, W. D. *The Connection Machine.* MIT Press, Cambridge, MA, 1985.
6. Hakimi, S. L., and Amin, A. T. Characterization of the connection assignment of diagnosable systems. *IEEE Trans. comput. C* **23**(1) (1974), 86–88.
7. Hastad, J., Leighton, F. T., and Newman, M. Fast computations using faulty hypercubes. *Proc. 21st Annual ACM Symp. Theory of Comput.* 1989, pp. 251–284.
8. Hayes, J. P., Mudge, T. N., and Stout, Q. F. Architecture of a hypercube supercomputer. *Proc. 1986 Int. Conf. Parallel Processing.* Aug. 1986, pp. 653–660.
9. Intel Corporation. *iPSC System Overview.* Jan. 1986.
10. Kuhl, J., and Reddy, S. Distributed fault-tolerance for large multiprocessor systems. *Proc. 7th Int. Symp. Comput. Architecture.* 1980, pp. 23–30.
11. Lee, T. C. Quick recovery of embedded structures in hypercube computers. *Proc. 5th Distributed Memory Computing Conf.* Apr. 1990, pp. 1426–1435.
12. Preparata, F. P., Metze, G., and Chien, R. T. On the connection assignment problem of diagnosable systems. *IEEE Trans. Electron. Comput. EC* **16**(12)(1967), 848–854.
13. Raghavendra, C. S., Yang, P.-J., and Tien, S.-B. Free dimensions—An efficient approach to achieving fault tolerance in hypercubes. *Proc. 22nd Int. Symp. Fault-Tolerant Comput.* July 1992, pp. 170–177.
14. Seitz, C. L. The Cosmic Cube. *Comm. ACM* **28**(1)(1985), 22–23.
15. Whitney, H. Congruent graphs and the connectivity of graphs, *Amer. J. Math.* **54,** (1932), 150–168.

DAJIN WANG received the B.Eng. from Shanghai University of Science and Technology, Shanghai, China, in 1982, and the M.S. and the Ph.D., both in computer science, from Stevens Institute of Technology, Hoboken, New Jersey, in 1986 and 1990, respectively. Since then, he has been with the faculty of the Department of Mathematics and Computer Science at Montclair State University, Upper Montclair, New Jersey, where he is currently an associate professor. His research interests include fault tolerant computing, algorithmic robotics, and parallel processing.

ZHONGXIAN WANG received the B.S. from Taiyuan University of Science and Technology, Taiyuan, China, in 1976, the M.S. from Harbin Institute of Technology, Harbin, China, in 1983, and the Ph.D. from Temple University, Philadelphia, in 1990. He is with the faculty of the Department of Information and Decision Sciences at Montclair State University, Upper Montclair, New Jersey, where he is currently an assistant professor. His research interests include optimization algorithms, synergy of automation, and operations research.