A Fault-Tolerant Routing Algorithm for 3-D Torus Interconnection Networks

Jehad Al-Sadi[1], Khaled Day [2], Mohamed Ould-Khaoua[3]
[1]Department of Computer Science, Zarka Private University, Jordan
[2]Department of Computer Science, Sultan Qaboos University, Sultanate of Oman
[3]Department of Computing Science, University of Glasgow, UK

Abstract: *This paper describes a new fault-tolerant routing algorithm for 3-D tori using the concept of "probability vectors". To compute these vectors, a node determines first its faulty set, which represents the set of all its neighbouring nodes that are faulty or unreachable due to faulty links. Each node then calculates a probability vector, where the l^{th} element represents the probability that a destination node at distance l cannot be reached through a minimal path due to a faulty node or link. The probability vectors are used by all the nodes to achieve an efficient fault-tolerant routing in the network. An extensive performance evaluation conducted in this study reveals that the proposed algorithm exhibits good fault-tolerance properties in terms of the achieved percentage of reachability and routing distances.*

Keywords: *Multicomputers, interconnection networks, 3-D tori, fault-tolerant routing, performance evaluation.*

Received April 27, 2003; accepted May 12, 2003

1. Introduction

3-D tori are one of the most common networks for multicomputers due to their desirable properties, such as ease of implementation and ability to reduce message latency by exploiting communication locality found in many parallel applications. 3-D torus; as a member of the k-ary n-cube networks family; possess a 3-dimensional grid structure with k nodes in each dimension such that every node is connected to its neighbouring nodes in each dimension by direct channels. The three most popular and widely studied instances of k-ary n-cubes are the hypercube (where k=2), the 2-D torus (where n=2), and the 3-D torus (where n=3). The hypercube has been used in early multicomputers such as the iPSC/2 [20] and iPSC/860 [24] while the torus has been adopted in recent systems like the J-Machine [19], CRAY T3D [14] and CRAY T3E [3].

A routing algorithm specifies how a message selects a path to cross from source to destination, and has great impact on network performance. Routing in fault-free networks has been extensively studied in the past [9, 10, 12, 17, 18, 22]. As the network size scales up the probability of processor and link failure also increases. It is therefore essential to design fault-tolerant routing algorithms that allow messages to reach their destinations even in the presence of faulty components (links and nodes). Existing fault-tolerant routing algorithms, discussed mainly in the context of the hypercube topology [4, 6, 13, 15, 16, 21], have assumed that a node knows either only the status of its neighbours (such a model is called local-information-based) [4, 6, 13] or the status of all nodes (global-information-based) [5, 23]. Local-information-based routing yields sub-optimal routes (if not routing failure) due to the insufficient information upon which the routing decisions are made. Global-information-based routing can achieve optimal or near optimal routing, but often at the expense of high communication overhead to maintain up-to-date network-wide fault information. The main challenge is therefore to devise a simple and efficient way of representing limited global fault information that allows optimal or near-optimal fault-tolerant routing.

There have recently been a number of studies reported in the literature that have described limited-global-information-based fault-tolerant routing algorithms. Most of these algorithms, however, have been developed for the hypercube [1, 7, 16, 25, 26]. As a result, little work has considered the other versions of the k-ary n-cubes, such as 3-D tori. In fact, most of the existing research on k-ary n-cubes has dealt with the practical and implementation issues associated with fault-tolerant routing [9, 10, 11, 12]. There has been hardly any study that investigates the *topological* properties of 3-D torus for the provision of efficient fault-tolerant routing algorithms.

Recently, the probability vectors have been proposed as a new framework for designing efficient limited-global information-based fault-tolerant routing algorithms [1, 2]. The authors in [2] have shown how the concept of probability vectors could be used to design a fault-tolerant routing for the hypercube networks, that has been shown to outperform existing algorithms, such as the safety vectors [25]. The study in [1, 2] have also argued that one of the main features of the probability vectors is their generality in that they could be applied to wide range of network topologies. This paper demonstrates how the probability vectors could be adapted to develop an efficient fault-tolerant algorithm for the well-known 3-D tori. The new algorithm uses the "probability vectors" to considerably reduce the storage requirement for maintaining fault information, compared to global-Information-based algorithms [5, 23]. In the proposed algorithm, each node *A* starts by determining the set of faulty or unreachable neighbours. Then each node *A* calculates its probability vector $P^A = (P_1^A, \ldots, P_{3\lfloor k/2 \rfloor}^A)$. The l^{th} element, P_l^A, of the probability vector represents the probability that a destination node at distance *l* from *A* cannot be reached from *A* using a minimal path due to faulty nodes and links. An extensive analysis is performed in this study to assess the performance of the proposed algorithm. The results presented here reveal that the new algorithm performs near optimal routing for practical values of the numbers of faulty nodes. Moreover, the results reveal that the algorithm exhibits good performance levels in terms of the achieved routing distances and percentages of reachability even when there exist a large number of faulty nodes.

The remainder of the paper is organised as follows. Section 2 reviews some background information (preliminaries and notation) that will be useful for the subsequent sections. Section 3 presents the proposed fault-tolerant algorithm for the 3-D torus. Section 4 presents an analytical study of the proposed algorithm. Section 5 conducts a performance evaluation of the new algorithm through simulation experiments. Section 6 concludes this study.

2. Preliminaries and Notation

The 3-D torus, Q_3^k, is an undirected graph with k^3 vertices (nodes). Each node *A* is labeled in the form $A = a_2, a_1, a_0$, where $0 \le a_i < k$. Two nodes $A = a_2, a_1, a_0$ and $B = b_2, b_1, b_0$ are joined by a link if, and only if, there exists i, $0 \le i < 3$, such that $a_i = b_i \pm 1$ (mod *k*) and $a_j = b_j$ for $i \ne j$. For the sake of clarity, we will omit writing *mod k* in similar expressions in the remainder of our discussion. Q_3^k has a degree of 6 and diameter $3\lfloor k/2 \rfloor$. The shortest path between nodes *A* and *B* is equal to their *Lee distance* given by

$$d_L(A,B) = \sum_{i=0}^{2} w_i \text{ , where}$$

$$w_i = \min_{0 \le i < 3} (|a_i - b_i|, k - |a_i - b_i|)$$

The two neighbours of a node *A*, along the i^{th} dimension are denoted as $A^{(i+)}$ and $A^{(i-)}$. Therefore, node *A* has six neighbours, two neighbours along each dimension i, $0 \le i < 3$. The *Hamming distance* between two nodes *A* and *B*, denoted *H* (*A*, *B*), is the number of dimensions at which their labels differ. A path between nodes *A* and *B* is an *optimal or minimal path*, if its length is equal to $d_L(A,B)$, i.e. the path has the minimum distance between *A* and *B*. When $a_i \ne b_i$, a neighbour $A^{(i\pm)}$ is called a *preferred neighbour* of *A* for the routing from *A* to *B* if $d_L(A^{(i\pm)}, B) = d_L(A, B) - 1$. We say in this case that $i\pm$ is a *preferred direction*. A minimal path can be obtained by performing a preferred direction move at every routing step. If a_i ^{1} b_i, a neighbour $A^{(i\pm)}$ such that $d_L(A^{(i\pm)}, B) \ge d_L(A, B)$ is called a *spare neighbour*. Neighbours other than preferred or spare are called *disturb neighbours*. For routing from *A* to *B*, a disturb neighbour $A^{(i\pm)}$ of *A* corresponds to the case $a_i = b_i$ and therefore the i^{th} digit is disturbed. Routing through a disturb neighbour increases the total routing distance by at least two over the minimum distance. Routing through a spare neighbour increases the total routing distance by at least one over the minimum distance. With respect to routing from node *A* to node *B*, a node *T* is called a preferred *transit node* if $d_L(T, B) < d_L(A,B)$.

We make the following assumptions for the purpose of the present study. Similar assumptions have been made in earlier related works, e.g. [9, 22, 25].

a. A faulty 3-D torus contains faulty nodes and/or links. The fault pattern remains fixed for the duration of calculating the probability vectors.
b. Each node can determine the status of its own links and the status of its neighbouring nodes.

3. The Proposed Probability-Based Fault-Tolerant Routing Algorithm

Our proposed fault-tolerant routing algorithm uses the concept of probability vectors. The probability vector of a node *A* is denoted by $P^A = (P_1^A, \ldots, P_{3\lfloor k/2 \rfloor}^A)$ where P_l^A represents the probability that a destination node at Lee distance *l* cannot be reached from node *A* using a minimal path due to faulty nodes and links. To calculate its probability vector, node *A* starts by determining the faulty set F_A, which comprises those neighbouring nodes that are either faulty or unreachable from *A* due to faulty links. After determining F_A, node *A* then calculates its probability vector $P^A = (P_1^A, \ldots, P_{3\lfloor k/2 \rfloor}^A)$ through $3\lfloor k/2 \rfloor$-1 exchanges of information with its neighbours (defined below). The probability vectors are used by all the nodes to perform efficient fault-tolerant routing in the network.

Definition 1: The faulty set F_A of a node *A* is defined as $F_A = \bigcup_{0 \le i < 3} f_A^i$, where f_A^i is given by

$$f_A^i = \begin{cases} \{A^{(i\pm)}\} & \text{if } A^{(i\pm)} \text{ is faulty or link}(A, A^{(i\pm)}) \text{ is faulty} \\ f & \text{Otherwise} \end{cases} \quad (1)$$

The l^{th} element P_l^A of the probability vector, P^A, denotes the probability that a destination at Lee distance l from A is not minimally reachable, i.e. reachable using a minimal path, from A. Since node A has $|F_A|$ faulty or unreachable immediate neighbours, and only one of the 6 edges incident from A constitutes a minimal path to a specific destination at Lee distance one, the first element of the probability vector, P_1^A, is given by

$$P_1^A = \frac{|F_A|}{6} \qquad (2)$$

In order to compute the other elements P_l^A, $l \geq 2$, let $R_l^{A^{(i)}}$ be the probability that a destination at Lee distance l from A is minimally reachable via its neighbour $A^{(i\pm)}$. The probability P_l^A, $l \geq 2$, satisfies the relations:

$$P_l^A \leq \prod_{i=0}^{2} (1 - R_l^{A^{(i\pm)}}), \text{ where}$$

$$R_l^{A^{(i\pm)}} \geq \begin{cases} 0 & \textit{if node } A^{(i\pm)} \textit{ is faulty} \\ \sum_{\mathrm{h=1}}^{\mathrm{min(l,n)}} \frac{h}{6}(1 - P_{l-1}^{A^{(i\pm)}}) & \textit{Otherwise} \end{cases} \qquad (3)$$

When node A has to route a message M towards its destination B it applies the probability vectors-based routing algorithm, referred to here as "PV_Routing", outlined in Figure. 1. If the route is through a preferred neighbour, $A^{(i\pm)}$, then the associated least expected routing distance is given by

$$Pr = l(1 - P_{l-1}^{A(i\pm)}) + (l+2)P_{l-1}^{A(i\pm)}$$

where $P_{l-1}^{A^{(i\pm)}}$ is the probability that a minimal path via the preferred neighbour $A^{(i\pm)}$ to a destination at Lee distance l is faulty. On the other hand, if the route is through a spare or disturb neighbour, $A^{(j\pm)}$, then the least expected routing distance is computed as

$$Sp = (l+2)(1 - P_{l+1}^{A(j\pm)}) + (l+4)P_{l+1}^{A(j\pm)}$$

The following text is an outline of the proposed PV_Routing fault-tolerant algorithm that node A uses to determine a path to route a message towards its destination B.

```
Algorithm PV_Routing (M: message; A,S,B: node)
/* Called by node A to route the message M initiated at source node S towards its destination node B */
if  A=S then M.Route_distance=0;
l = Lee distance between A and B;
if M.Route_distance= l+(k-2) x no_faulty_nodes then
{
M.Route_distance= M.Route_distance + 1;
if B is a reachable neighbour then deliver M to B; exit;
/* destination reached */
 Let $A^{(i\pm)}$ be a reachable preferred neighbour with least $P_{l-1}^{A^{(i\pm)}}$ value;
 $Pr = l(1 - P_{l-1}^{A(i\pm)}) + (l+2)P_{l-1}^{A(i\pm)}$; /* least expected distance through $A^{(i\pm)}$ */
 Let $A^{(j\pm)}$ be a reachable spare neighbour with least $P_{l+1}^{A^{(j\pm)}}$ value;
 $Sp = (l+2)(1 - P_{l+1}^{A(j\pm)}) + (l+4)P_{l+1}^{A(j\pm)}$;  /* least expected distance through $A^{(j\pm)}$ */
 if $ $A^{(i\pm)}$ and ( ($ $A^{(j\pm)}$ and $Pr \leq Sp$) or (~$ $A^{(j\pm)}$ ) ) then send M to $A^{(i\pm)}$;
 else if $ $A^{(j\pm)}$ and ( ($ $A^{(i\pm)}$ and $Pr > Sp$) or (~ $ $A^{(i\pm)}$ ) ) then send M to $A^{(j\pm)}$;
        else failure      /* destination unreachable */
} else Detect_Looping
end. /* algorithm */
```

Example 1: Consider the 3-D torus with k = 3 and five faulty nodes shown in Figure 1 (faulty nodes are drawn in dark color). Table 1 shows the corresponding faulty set and probability vectors associated with each node *A*. To route a message from the source node (200) to the destination node (222), first node (200) has two preferred neighbours (220) and (202), but since node (220) is faulty, the proposed routing algorithm will route to node (202) as an intermediate node. Node (202) has one spare neighbour (long cycle neighbour) (212), but the algorithm routes the message via the preferred dimension to its destination node (222).

Notice from the description of the proposed PV_Routing algorithm that looping can be detected if the routing distance exceeds the specified limit (Lee distance plus f(k-2) where f is the number of faulty nodes). Since each faulty node may cause a derouting and an increase in the routing distance by a value ranging between 1 and k-2, the maximum increase in the routing distance should not exceed f(k-2). The algorithm can be improved to minimize the effects of looping. Since looping occurs when a destination is not reachable from a source we can add the destination node to the faulty set of the node that detected the looping. When looping occurs ($3^{\lfloor k/2 \rfloor}$-1) exchanges of information between all neighbours is then initiated to propagate the new information among reachable nodes in the whole 3-D torus.

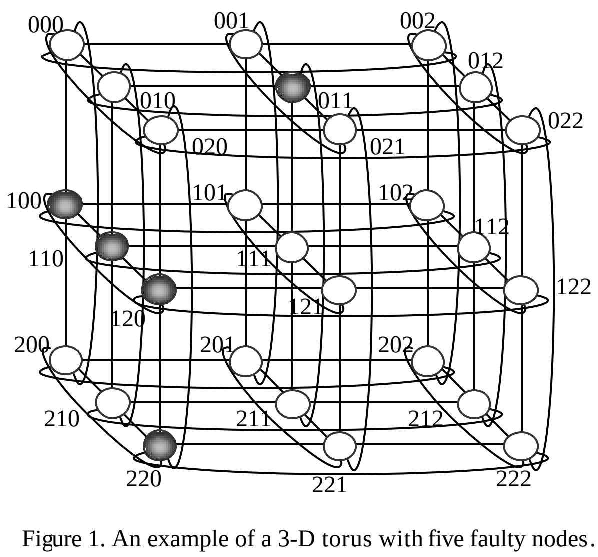


Figure 1. An example of a 3-D torus with five faulty nodes.

Table 1. The faulty sets and probability vectors for a 3-D torus with 5 faulty nodes.

Node A	(000)	(001)	(002)	(010)	(011)	(012)	(020)	(021)	(022)
F^A	{100}	{011}	{}	{011,110}	Faulty	{011}	{120,220}	{011}	{}
P_1^A	0.167	0.167	0.000	0.333	Faulty	0.167	0.333	0.167	0.000
P_2^A	0.086	0.050	0.029	0.151	Faulty	0.49	0.113	0.066	0.039

Node A	(100)	(101)	(102)	(110)	(111)	(112)	(120)	(121)	(122)
F^A	Faulty	{100}	{100}	Faulty	{011,110}	{110}	Faulty	{120}	{120}
P_1^A	Faulty	0.167	0.167	Faulty	0.333	0.167	Faulty	0.167	0.167
P_2^A	Faulty	0.066	0.050	Faulty	0.116	0.066	Faulty	0.077	0.058

Node A	(200)	(201)	(202)	(210)	(211)	(212)	(220)	(221)	(222)
F^A	{100,220}	{}	{}	{110,220}	{011}	{}	Faulty	{220}	{220}
P_1^A	0.333	0.000	0.000	0.333	0.167	0.000	Faulty	0.167	0.167
P_2^A	0.097	0.039	0.028	0.130	0.065	0.039	Faulty	0.058	0.043

4. Performance Analysis

This section analyses some performance properties of the proposed PV_routing algorithm in terms of the achieved minimum and average routing distances for various sizes of the 3-D torus networks. In the remainder of the paper, we assume that there are f faulty nodes in the network, and that all the N nodes are equally likely to be faulty. Furthermore, we assume that the source and destination nodes are non-faulty. Let us now start by deriving a lower bound on the probability of minimum distance routing using the new algorithm.

4.1. A Lower Bound on the Probability of Minimum Distance Routing

For any two nodes at Lee distance l, $1 \le l \le 3\lfloor k/2 \rfloor$, and Hamming distance h, $1 \le h \le 3$, the 3-D torus is known [8] to embed a family p of 6 node-disjoint paths of the following lengths:

h paths of length l,

$6 - 2h$ paths of length l+2, and

h paths of length l+4

Assume there exists a "hypothetical" routing algorithm R that attempts to route along a non-faulty path from the family p of shortest possible lengths before considering other paths. The following theorem provides a lower bound on the probability of minimum distance routing achieved by the algorithm R.

Theorem 1: For any source A and destination B at Lee distance l, $1 \le l \le 3\lfloor k/2 \rfloor$, and Hamming distance h, $1 \le h \le 3$, the routing algorithm R routes from A to B on a path of length at most l + 4 with probability of at least $1 - P_l \cdot P_{l+1} \cdot P_{l+4}$, where

$$P_l = \left(1 - (1-q)^l\right)^h \tag{4}$$

$$P_{l+2} = \left(1 - (1-q)^{l+2}\right)^{6-2h} \tag{5}$$

$$P_{l+4} = \left(1 - (1-q)^{l+4}\right)^h \tag{6}$$

$$q = \frac{f}{k^3} = \frac{f}{N} \tag{7}$$

Proof: Let P_l, $1 \le l \le 3\lfloor k/2 \rfloor$, be the probability that all node-disjoint paths in p of length l are faulty. Such a path is faulty if at least one of its l nodes (other than the source node) is faulty. A node is faulty with probability $q = f/N$ since there are f faulty nodes and all the N nodes in the network are equally likely to be faulty. Therefore a path of length l from p is faulty with probability $1 - (1-q)^l$, and hence $P_l = \left(1 - (1-q)^l\right)^h$. Similar analysis yields the expressions for P_{l+2} and P_{l+4}. Therefore at least one of the 6 paths of p is non-faulty with probability $1 - P_l \cdot P_{l+1} \cdot P_{l+4}$. □

The PV_Routing algorithm attempts to route through a neighbour that has the highest probability of minimum distance routing. The new algorithm keeps all options open and may select from any of the possible paths. As a result, it does not have any preference for a particular family of paths as does algorithm R in Theorem 1. It is therefore expected that PV_Routing will perform at least as good as algorithm R. In other words, the probability that PV_Routing routes from a source A to a destination B at Lee distance l on a minimum distance path with at least the probability 1- $P_l \cdot P_{l+2} \cdot P_{l+4}$.

Claim 1: PV_Routing routes messages between a given pair of nodes at Lee distance l, $1 \le l \le 3\lfloor k/2 \rfloor$, and Hamming distance h, $1 \le h \le 3$, on a minimum distance path with at least the probability $1 - P_l \cdot P_{l+2} \cdot P_{l+4}$.

This claim is verified experimentally by analysing the performance of the proposed algorithm in order to measure the path lengths against the number of faulty nodes in the network. To this end, simulation experiments have been carried out over an 3-D torus Q_3^8 with 512 nodes with different random distributions of faulty nodes. We started our experiments with a non-faulty 3-D torus and then the number of faulty nodes was increased gradually up to 70% of the network size with random fault distribution. Paths from every node A to all destinations at Lee distance 6 and Hamming distance 3 (as an average Lee and Hamming distances) were selected. Figure 2 shows both the calculated (based on claim 1) and the measured probability of minimum distance routing against the number of faulty nodes in the Q_3^8 when the Lee distance is l=6 and the Hamming distance is h=3. Other simulation experiments have been carried out over the Q_3^8 with a fixed number of faulty nodes 153 (30% of the nodes) with different random distributions. A total of 300,000 source-destination pairs were randomly selected. Table 2 contains both the calculated and measured probabilities of minimum distance routing for different Lee and Hamming distances. Both Figure 3 and Table 2 confirm that the probability of minimum distance routing for PV_Routing is always better than the corresponding probability for the hypothetical routing algorithm R. This shows that the probability that PV_Routing routes from a source A to a destination B at Lee distance l on a minimum length path is at least $1 - P_l P_{l+2} P_{l+4}$.

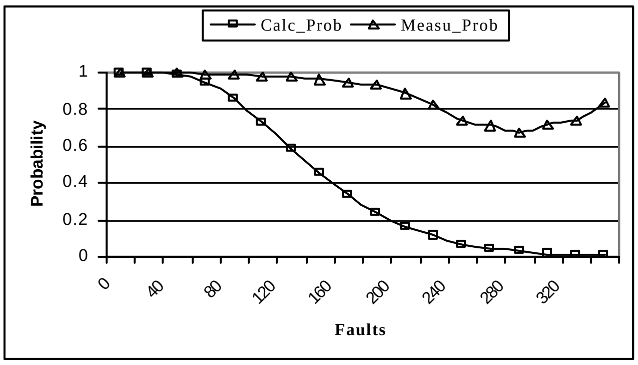


Figure 3. The calculated and measured probability of minimum distance routing against the number of faulty nodes in the 3-D torus Q_3^8.

4.2. The Average Routing Distance in the 3-D Torus

In order to evaluate the average routing distance of PV_Routing, we define a hypothetical class of probabilistic routing algorithms. We then evaluate the performance of these algorithms with the aim of deriving bounds on the performance of PV_Routing.

Table 2. Calculated and measured probability of minimum distance routing for a fixed number of faulty nodes (30% faulty nodes) in the 3-D torus Q_3^8.

Lee Dist.	Hamming Dist.	Calculated Prob.	Measured Prob.
1	1	0.954	1
2	1	0.852	0.922
2	2	0.885	0.970
3	1	0.714	0.888
3	2	0.751	0.925
3	3	0.783	0.962
4	1	0.570	0.877
4	2	0.604	0.889
4	3	0.636	0.942
5	2	0.467	0.894
5	3	0.495	0.908
6	2	0.351	0.885
7	2	0.258	0.873
8	2	0.187	0.877
8	3	0.200	0.878
9	3	0.144	0.862
10	3	0.103	0.865
11	3	0.073	0.870
12	3	0.052	0.892

Definition 2: A routing algorithm is called a Probabilistic Routing Algorithm (or PRA for short) if the routing decisions are based on maximising the probability of minimum distance routing when selecting a node from the faulty-free neighbours.

The following assumptions are made to simplify the analysis of the PRA algorithm and to derive bounds on the performance of the PV_Routing algorithm.

a. In selecting the next move, the neighbours are considered in the following order: preferred on the first dimension, preferred on the second dimension, preferred on the third dimension, spare on the first dimension, spare on the second dimension, and spare on the third dimension.
b. After f spare routing moves, the message is discarded to avoid looping.

Lemma 1: PV_Routing is a PRA.

Proof: Since the PV_Routing algorithm decisions are based on maximising the probability of minimum distance routing when selecting a node from the faulty-free neighbours, and it satisfies the above conditions (*a*.) and (b.), then PV_Routing is a PRA. □

The PV_Routing algorithm routes messages depending on the probabilistic value P_l^A used as a base of the routing decisions to destinations at Lee distance l. The maximum number of spare moves a path is expected to pass through using PV_Routing is f. The total path length in such a case is the Lee distance between source and destination plus $2f$. If the path exceeded f spare moves then looping must have occurred and the message is then discarded.

We now derive an expression for the average routing distance in the PRA algorithm. Since the 3-D torus has symmetric network topology, we will focus, without loss of generality, our discussion on a particular source node, *S*. We will use the following notation during the derivation:

- $P_{l_1,l_2,l_3,s}$: probability of making exactly *s* spare moves when routing between the source node *S* and a destination with Lee distance components (l_1, l_2, l_3), where $l = l_1 + l_2 + l_3$, and l_1, l_2 and l_3 are the Lee distance across the first, second, and third dimension, respectively.
- $\overline{D}_l$: average routing distance to a destination at Lee distance *l* from the fixed source node *S*.
- $\overline{D}_{l_1,l_2,l_3}$: average routing distance to a destination with Lee distance components (l_1, l_2, l_3).
- w_{l_1,l_2,l_3}: ratio of the number of nodes with Lee distance components (l_1, l_2, l_3) to the number of nodes at Lee distance $l = l_1 + l_2 + l_3$ from the fixed source node *S*.
- N_l: number of nodes at Lee distance *l* from the fixed source node S.

lemma 2: w_{l_1,l_2,l_3} is given by

$$w_{l_1,l_2,l_3} = \begin{cases} \dfrac{2}{N_l} & l_1 > 0 \text{ xor } l_2 > 0 \text{ xor } l_3 > 0 \\ \dfrac{4}{N_l} & l_1 = 0 \text{ or } l_2 = 0 \text{ or } l_3 = 0 \\ \dfrac{8}{N_l} & l_1 > 0 \text{ and } l_2 > 0 \text{ and } l_3 > 0 \end{cases} \tag{8}$$

Proof: For a given l_1, l_2, and l_3, $0 \le l_1, l_2, l_3 \le \lfloor k/2 \rfloor$, and a fixed source node $S = (x_1, x_2)$ in the 3-D torus, if l_1 ? 0, l_2 ? 0 and l_3 ? 0 then there are eight possible destination nodes with Lee distance components (l_1, l_2, l_3) from *S*. These are: (x_1+l_1, x_2+l_2, x_3+l_3), (x_1+l_1, x_2-l_2, x_3+l_3), (x_1-l_1, x_2+l_2, x_3+l_3), (x_1-l_1, x_2-l_2, x_3+l_3), (x_1+l_1, x_2+l_2, x_3-l_3), (x_1+l_1, x_2-l_2, x_3-l_3), (x_1-l_1, x_2+l_2, x_3-l_3), and (x_1-l_1, x_2-l_2, x_3-l_3). If however l_1=0 or l_2=0 or l_3=0, then there are four such possible destinations. If l_1>0 or l_2>0 or l_3>0, then there are only two such possible destinations. Furthermore, the number of destinations N_l at a given Lee distance *l* from the source *S* is given by

$$N_l = \begin{cases} \displaystyle\sum_{l_1=0}^{\lfloor k/2 \rfloor} \sum_{l_2=0}^{\lfloor k/2 \rfloor} \sum_{l_3=0}^{\lfloor k/2 \rfloor} 1 & l_1 + l_2 + l_3 = l \text{ and } l > \lfloor k/2 \rfloor \\ 6 + 12(l-1) + 8\displaystyle\sum_{i=0}^{l-2} i & l_1 + l_2 + l_3 = l \text{ and } l \le \lfloor k/2 \rfloor \end{cases} \tag{9}$$

□

Lemma 3: $\overline{D}_{l_1,l_2,l_3}$ is given by

$$\overline{D}_{l_1,l_2,l_3} = \sum_{s=0}^{f} (l_1 + l_2 + l_3 + 2s) P_{l_1,l_2,l_3,s}$$

$$\begin{aligned} P_{l_1,l_2,l_3,s} = {} & (1-p)P_{l_1-1,l_2,l_3,s} + p(1-p)P_{l_1,l_2-1,l_3,s} + \\ & p^2(1-p)P_{l_1,l_2,l_3-1,s} + p^3(1-p)P_{l_1+1,l_2,l_3,s-1} \\ & + p^4(1-p)P_{l_1,l_2+1,l_3,s-1} + p^5(1-p)P_{l_1,l_2,l_3+1,s-1} \end{aligned}$$

The probability $P_{l_1,l_2,s}$ satisfies the following boundary conditions, as in equation (10).

$$P_{l_1,l_2,l_3,s} = \begin{cases} 1 & l_1 = 1, l_2 = 0, l_3 = 0, s = 0 \\ 1 & l_1 = 0, l_2 = 1, l_3 = 0, s = 0 \\ 1 & l_1 = 0, l_2 = 1, l_3 = 1, s = 0 \\ (1-p)^{l_1-1} & l_1 > 0, l_2 = 0, l_3 = 0, s = 0 \\ (1-p)^{l_2-1} & l_1 = 0, l_2 > 0, l_3 = 0, s = 0 \\ (1-p)^{l_3-1} & l_1 = 0, l_2 = 0, l_3 > 0, s = 0 \\ (1-p)P_{l_1-1,l_2,l_3,0} + p(1-p)P_{l_1,l_2-1,l_3,0} + p^2(1-p)P_{l_1,l_2-1,l_3,0} & l_1 > 0, l_2 > 0, l_3 > 0, s = 0 \\ 0 & l_1 = 1, l_2 = 0, l_3 = 0, s > 0 \\ 0 & l_1 = 0, l_2 = 1, l_3 = 0, s > 0 \\ 0 & l_1 = 0, l_2 = 0, l_3 = 1, s > 0 \\ (1-p)P_{l_1-1,0,0,s} + p(1-p)P_{l_1+1,0,0,s-1} + p^2(1-p)P_{l_1,1,0,s-1} & \\ \quad + p^3(1-p)P_{l_1,0,1,s-1} & 1 < l_1 < \lfloor k/2 \rfloor, l_2 = 0, l_3 = 0, s > 0 \\ (1-p)P_{l_1-1,0,0,s} + p(1-p)P_{l_1,1,0,s-1} + p^2(1-p)P_{l_1,0,1,s-1} & l_1 = \lfloor k/2 \rfloor, l_2 = 0, l_3 = 0, s > 0 \\ (1-p)P_{0,l_2-1,0,s} + p(1-p)P_{1,l_2,s-1} + p^2(1-p)P_{0,l_2+1,0,s-1} & \\ \quad + p^3(1-p)P_{0,l_2,1,s-1} & l_1 = 0, 1 < l_2 < \lfloor k/2 \rfloor, l_3 = 0, s > 0 \\ (1-p)P_{0,0,l_3-1,s} + p(1-p)P_{1,0,l_3,s-1} + p^2(1-p)P_{0,1,l_3,s-1} & \\ \quad + p^3(1-p)P_{0,0,l_3+1,s-1} & l_1 = 0, l_2 = 0, 1 < l_3 < \lfloor k/2 \rfloor, s > 0 \\ (1-p)P_{0,l_2-1,0,s} + p(1-p)P_{1,l_2,0,s-1} + p^2(1-p)P_{0,l_2,1,s-1} & l_1 = 0, l_2 = \lfloor k/2 \rfloor, l_3 = 0, s > 0 \\ \left.\begin{aligned} & (1-p)P_{l_1-,1,l_2,l_3,s} + p(1-p)P_{l_1,l_2-1,l_3,s} + p^2(1-p)P_{l_1,l_2,l_3-1,s} \\ & + p^3(1-p)P_{l_1+1,l_2,l_3,s-1} + p^4(1-p)P_{l_1,l_2+1,l_3,s-1} + p^5(1-p)P_{l_1,l_2,l_3+1,s-1} \end{aligned}\right\} & 0 < l_1, l2, l_3 < \lfloor k/2 \rfloor, s > 0 \\ (1-p)P_{l_1-1,l_2,l_3,s} + p(1-p)P_{l_1,l_2-1,l_3,s} + p^2(1-p)P_{l_1,l_2,l_3-1,s} & l_1 = l_2 = l_3 = \lfloor k/2 \rfloor, s > 0 \end{cases} \tag{10}$$

Proof: Let $\overline{D}_{l_1,l_2,l_3}$ be the average routing distance between a given pair of nodes with Lee distance components (l_1,l_2,l_3). Since each spare move increases the routing distance by 2 hops, and since messages are discarded after making *f* spare moves, we can write $\overline{D}_{l_1,l_2,l_3}$ as

$$\overline{D}_{l_1,l_2,l_3} = \sum_{s=0}^{f}(l_1+l_2+l_3+2s)P_{l_1,l_2,l_3,s}$$ □

To make *s* spare moves when routing a message at Lee distance *l* from its destination, we distinguish the following cases based on the first made move:

a. A preferred move on the first dimension leading to a node with Lee distance components (l_1-1,l_2,l_3) from destination, and the remaining route must include *s* spare moves.
b. A preferred move on the second dimension leading to a node with Lee distance components (l_1,l_2-1,l_3) from destination, and the remaining route must include *s* spare moves.
c. A preferred move on the third dimension leading to a node with Lee distance components (l_1,l_2,l_3-1) from destination, and the remaining route must include *s* spare moves.
d. A spare move on the first dimension leading to a node with Lee distance components (l_1+1,l_2,l_3) from destination, and the remaining route must include s-1 spare moves.
e. A spare move on the second dimension leading to a node with Lee distance components (l_1,l_2+1,l_3) from destination, and the remaining route must include *s*-1 spare moves.
f. A spare move on the third dimension leading to a node with Lee distance components (l_1,l_2,l_3+1) from destination, and the remaining route must include *s*-1 spare moves.

It can be easily verified that $P_{1,0,0,0}=1, P_{0,1,0,0}=1, P_{0,0,1,0}=1, P_{1,0,0,s}=0$, $P_{0,1,0,s}=0$ and $P_{0,0,1,s}=0$ for all $s > 0$ since the source and destination nodes are both assumed to be non-faulty. For $l=l_1+l_2+l_3 \geq 2$, $P_{l_1,0,0,0}$ is the probability that a destination with Lee distance components $(l_1,0,0)$ is minimally reachable. This probability is equal to $(1-p)^{l_1-1}$ as this requires all l_1-1 preferred intermediate nodes to be non-faulty. Using similar arguments the probabilities $P_{0,l_2,0,0}, P_{0,0,l_3,0}$ and $P_{l_1,l_2,l_3,0}$ are obtained. For $0<l_1,l_2,l_3<\lfloor k/2 \rfloor$, and $s>0$, we therefore can write $P_{l_1,l_2,l_3,s}$ as

$$\begin{aligned}P_{l_1,l_2,l_3,s} &= (1-p)P_{l_1-1,l_2,l_3,s} + p(1-p)P_{l_1,l_2-1,l_3,s} + \\ &p^2(1-p)P_{l_1,l_2,l_3-1,s} + p^3(1-p)P_{l_1+1,l_2,l_3,s-1} \\ &+ p^4(1-p)P_{l_1,l_2+1,l_3,s-1} + p^5(1-p)P_{l_1,l_2,l_3+1,s-1}\end{aligned} \quad (11)$$

When the destination is at Lee distance $\lfloor k/2 \rfloor$ on one, two or three dimensions, then the first move can only be a preferred move on that dimension, as in equation (12). The results of Lemmas 2 and 3 are used to obtain the following theorem.

Theorem 2: For the PRA algorithm, the average routing distance, $\overline{D}_l$, between a given pair of nodes at Lee distance *l* in the 3-D torus is given by

$$\overline{D}_l = \sum_{l_1=0}^{x}\sum_{l_2=0}^{x}\sum_{l_3=0}^{x} D_{l_1,l_2,l_3} \cdot w_{l_1,l_2,l_3},$$

where $l_1+l_2+l_3=l$ and $x=\min(l,\lfloor k/2 \rfloor)$

Claim 2: The average routing distance between two nodes at Lee distance *l* for PV_Routing in the 3-D torus is at most $\overline{D}_l$.

This claim is intuitively justified by the fact that PV_Routing makes routing decisions based on maximising the calculated probability of minimum distance routing while the PRA algorithm selects the first feasible move from the list: preferred on the first dimension, preferred on the second dimension, preferred on the third dimension, spare on the first dimension, spare on the second dimension, and spare on the third dimension. Since the PV_routing and PRA algorithms are based on similar probabilistic nature, we expect them to perform similarly in terms of the achieved average routing distance.

$$P_{l_1,l_2,l_3,s} = \begin{cases} (1-p)P_{l_1-1,l_2,l_3,s} + p(1-p)P_{l_1,l_2-1,l_3,s} + p^2(1-p)P_{l_1,l_2,l_3-1,s} & \\ + p^3(1-p)P_{l_1,l_2+1,l_3,s-1} + p^4(1-p)P_{l_1,l_2,l_3+1,s-1} & l_1=\lfloor k/2 \rfloor,\ 0<l_2,l_3<\lfloor k/2 \rfloor \\ (1-p)P_{l_1-1,l_2,l_3,s} + p(1-p)P_{l_1,l_2-1,l_3,s} + p^2(1-p)P_{l_1,l_2,l_3-1,s} & \\ + p^3(1-p)P_{l_1+1,l_2,l_3,s-1} + p^4(1-p)P_{l_1,l_2,l_3+1,s-1} & l_2=\lfloor k/2 \rfloor,\ 0<l_1,l_3<\lfloor k/2 \rfloor \\ (1-p)P_{l_1-1,l_2,l_3,s} + p(1-p)P_{l_1,l_2-1,l_3,s} + p^2(1-p)P_{l_1,l_2,l_3-1,s} & \\ + p^3(1-p)P_{l_1+1,l_2,l_3,s-1} + p^4(1-p)P_{l_1,l_2+1,l_3,s-1} & l_3=\lfloor k/2 \rfloor,\ 0<l_1,l_2<\lfloor k/2 \rfloor \\ (1-p)P_{l_1-1,l_2,l_3,s} + p(1-p)P_{l_1,l_2-1,l_3,s} + p^2(1-p)P_{l_1,l_2,l_3-1,s} & l_1=l_2=l_3=\lfloor k/2 \rfloor \end{cases} \quad (12)$$

To support this intuitive claim, we have compared the results obtained using the above-derived expressions against those obtained through simulation. We have first solved the equations related to w_{l_1,l_2,l_3}, $P_{l_1,l_2l_3,s}$, $\overline{D}_{l_1,l_2,l_3}$, and $\overline{D}_l$ given by Lemma 2, Lemma 3, and Theorem 2. These calculations yield the average routing distance vector $\overline{D}_A = (\overline{D}_1, \overline{D}_2, ..., \overline{D}_{n\lfloor k/2 \rfloor})$. We then performed experiments of our PV_Routing algorithm to measure experimentally the corresponding average routing distances vector.

Table 3 show results for the calculated and measured average routing distances, respectively, for the average routing distances in PV_Routing for different sizes of the 3-D torus Q_3^k, where the number of faulty nodes is 20% of the total network size. The experimental results and the analytical results are in close agreement with those obtained using simulation, demonstrating the accuracy of our above analytical derivation.

Table 3. The measured and calculated average routing distance for a fixed number of faulty nodes (20% faulty) in the 3-D torus of different sizes.

k	Lee Dist	Measured	Calculated
3	1	1	1
	2	2.082	2.043
	3	3.112	3.014
5	1	1	1
	2	2.210	2.203
	3	3.240	3.277
	4	4.246	4.233
	5	5.270	5.354
	6	6.279	6.347
7	1	1	1
	2	2.317	2.232
	3	3.478	3.408
	4	4.568	4.501
	5	5.554	5.573
	6	6.660	6.643
	7	7.680	7.693
	8	8.787	8.754
	9	9.792	9.726
9	1	1	1
	2	2.323	2.235
	3	3.434	3.426
	4	4.551	4.602
	5	5.791	5.709
	6	6.791	6.785
	7	7.840	7.875
	8	8.905	8.954
	9	10.075	10.012
	10	11.954	11.083
	11	12.157	12.159
	12	13.122	13.096

5. Experimental Performance Analysis

This section obtains experimentally three additional performance measures on the proposed PV_routing algorithm, namely deviation from optimality, unreachability, and looping. To this end, simulation experiments have been carried out over a 3-D tours Q_3^3 with 27 nodes with different random distributions of faulty nodes. We have started the experiments with a non-faulty 3-D tours Q_3^3 and then the number of faulty nodes was increased gradually up to 75% of the network size with random fault distributions. A total of 30,000 source-destination pairs where selected randomly at each run. The first two sets of results reported in Figures (4, 5). Before presenting the results, we define the following variables, which will be used to quantify some performance measures.

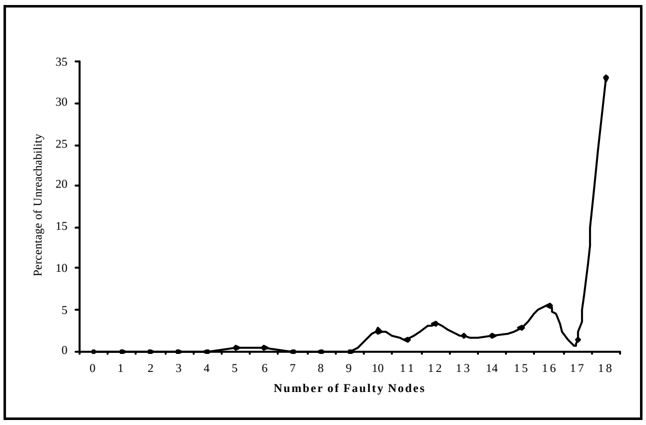
- *Total*: total number of generated messages.
- *Routing_Distance*: number of links crossed by a message.
- *Lee_Distance*: Lee distance between the message source and destination.
- *Fail_Count*: number of routing failure cases.
- *Looping_Count*: number of messages that cross a number of links beyond a maximum threshold before being discarded.

Using the above variables we propose the following three performance measures as the basis for studying the PV_Routing algorithm [21, 24].

- Average percentage of deviation from optimality $= \frac{1}{Total}\sum \frac{Routing\_Distance - Lee\_Distance}{Lee\_Distance} \times 100$
- Percentage of unreachability $= \frac{Fail\_Count}{Total} \times 100$
- Percentage of looping $= \frac{Looping\_Count}{Total} \times 100$

The average deviation from optimality indicates how close the achieved routing is to the minimal distance routing. The percentage of unreachability measures the percentage of messages that the algorithm failed to deliver to destination due to faulty components. The percentage of loops indicates the ratio of messages that failed to reach destinations due to network partitioning. We believe that these three measures give realistic indications on the performance of a fault-tolerant routing algorithm and are sufficient for the purpose of our current study.

Figure 3 reveals that PV_Routing achieves high reachability with low percentage of deviation from optimality. The deviation from optimality remains low as long as this number of faulty nodes does not exceed 50% of the total number of nodes, then it grows almost linearly with the number of faulty nodes. The proposed algorithm is capable of routing messages using optimal distance paths even when there are a large number of faulty components. This is due to the fact that the algorithm repeatedly chooses to route through areas of the network with the least number of faults in the neighbourhood by choosing to route to a preferred neighbour with the least probability that a destination at distance *l* from *A* is not minimally reachable from *A*. As a result, the algorithm tends to select paths that diverge from areas with high counts of faulty components. The result also reveals that the percentage of looping remains practically negligible when the percentage of faulty nodes is less than 40%.



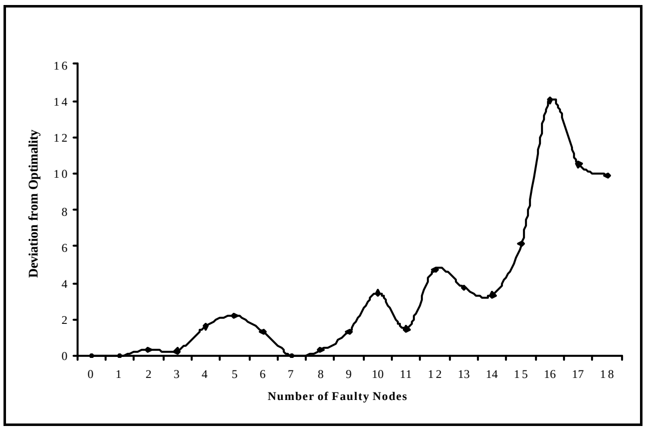
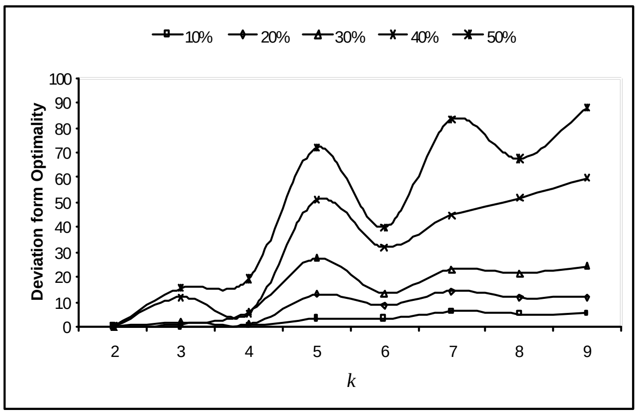


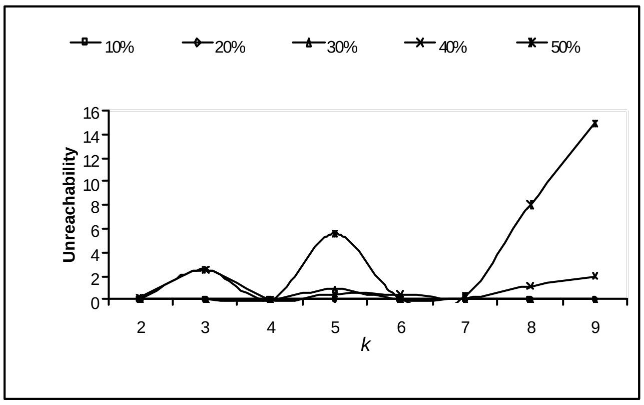
Figure 3. Percentage of deviation and unreachability in the proposed PV_Routing algorithm.

Another experiment was conducted to evaluate the performance behavior of our algorithm when the network size increases. For the sake of illustration, we have increased the value for k from 2 up to 9. For each network size, the algorithm has been tested by setting the percentage of faulty nodes to 10% of the network size, then to 20%, 30%, 40%, and 50% of the network size. At each run, a total of 30,000 source-destination pairs where selected randomly. The result presented in Figure 4 shows that the performance properties of the new fault-tolerant routing algorithm are not affected as the network size is scaled up. This reveals that the new algorithm possesses the nice property of maintaining good performance levels without imposing any restriction on the system size.

6. Conclusions

3-D tori are one of the most common networks for multicomputers due to their ease of implementation and ability to exploit communication locality found in many parallel applications. This study has first introduced the concept of “probability vectors”, and then used it to propose a new fault-tolerant routing algorithm for 3-D torus . As a first step in the new algorithm, each node A determines its view of the faulty set F_A of neighbouring nodes which are either faulty or unreachable from A. Equipped with these faulty sets node A calculates its probability vectors, P^A, by exchanging fault information with its reachable neighbours. An element P_l^A, $1 \le l \le 3\lfloor k/2 \rfloor$, of the vector represents the probability that a destination node at distance l cannot be reached from node A using a minimal path due to a faulty node or link along the path. Each node then uses the probability vectors to perform an efficient fault-tolerant routing in the network.





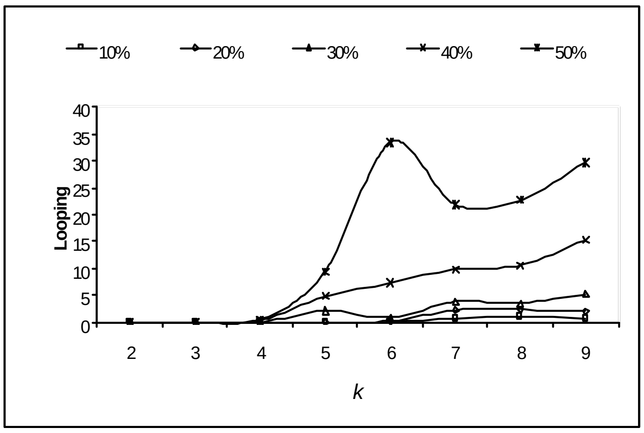


Figure 4. Percentage of deviation, unreachability, and looping in the proposed PV_Routing algorithm for different sizes of the 3-D torus (each curve represents a specific percentage of faulty nodes).

An analytical study has been presented to derive upper bounds on the average routing distance achieved by the proposed algorithm. An experimental performance analysis of the proposed algorithm using simulation experiments has also been reported. The results have revealed that the new algorithm provides good performance in terms of the routing distance and percentage of reachability even when the number of faulty nodes in the network is large. The results have also revealed that the algorithm maintains good performance levels as the network sizes scales up.

References

[1] Al-Sadi J., Day K., and Ould-Khaoua M., "Probability-based fault-tolerant routing in hypercubes," *in Proceedings of (Europar'2000), in Lecture Notes in Computer Science*, Springer-Verlag, Munich, pp. 935-938, 2000.

[2] Al-Sadi J., Day K., and Ould-Khaoua M., "Fault-tolerant routing in hypercubes using probability vectors," *Parallel Computing*, vol. 27, no. 10, pp. 1381-1399, 2001.

[3] Anderson E., Brooks J., Grassl C., and Scott S., "Performance of the Cray T3E multiprocessor," *in Proceedings of Supercomputing Conference*, San Jose, California, 1997.

[4] Chen M. S. and Shin K. G., "Adaptive fault-tolerant routing in hypercube multicomputers," *IEEE Trans. Computers*, vol. 39, no. 12, pp. 1406-1416, 1990.

[5] Chen M. S. and Shin K. G., "On hypercube fault-tolerant routing using global information," *in Proceedings of 4th Conf. Hypercube Concurrent Computers & Applications*, pp. 83-86, 1989.

[6] Chen M. S. and Shin K. G., "Depth-first search approach for fault-tolerant routing in hypercube multicomputers," *IEEE Trans. Parallel & Distributed Systems*, vol. 1, no. 2, pp. 152-159, 1990.

[7] Chiu G. M. and Chen K. S., "Fault-tolerant routing strategy using routing capability in hypercube multicomputers," *in Proceedings of Int. Conf. Parallel and Distributed Systems*, pp. 396-403, 1996.

[8] Day K. and Al-Ayyoub A., "Fault Diameter of *K*-ary n-cube Networks," *IEEE Trans. Parallel & Distributed Systems*, vol. 8, no. 9, pp. 903-907, 1997.

[9] Duato J., Yalamanchili S., and Ni L., *Interconnection networks: An engineering approach*, IEEE Computer Society Press, 1997.

[10] Gaughan P. T. and Yalamanchili S., "Adaptive routing protocols for hypercube interconnection networks," *IEEE Computer*, vol. 26, no. 5, pp. 12-24, 1993.

[11] Graham S. and Seidel S., "The cost of broadcasting on star graphs and *k*-ary hypercubes," *IEEE Trans. Computers*, vol. 42, no. 6, pp. 756-759, 1993.

[12] Gravano L. and Pifarre G., Berman P., and Sanz J., "Adaptive deadlock- and livelock-free routing with all minimal paths in torus networks," *IEEE Trans. Parallel & Distributed Systems*, vol.5, no.12, pp. 1233-1251, 1994.

[13] Gordon J. M. and Stout Q. F., "Hypercube message routing in the presence of faults," *in Proceedings of 3rd Conf. Hypercube Concurrent Computers and Applications*, Pasadena, California, pp. 251-263, 1988.

[14] Kessler R. E. and Schwarzmeier J. L., "CRAY T3D: A new dimension for Cray research," *in CompCon*, Houston, Texas, pp. 176-182, 1993.

[15] Lan Y., "A fault-tolerant routing algorithm in hypercubes," *in Proceedings of Int. Conf. Parallel Processing*, pp. 163-166, 1994.

[16] Lee T. C. and Hayes J. P., "A fault-tolerant communication scheme for hypercube computers," *IEEE Trans. Computers*, vol. 41, no. 10, pp. 1242-1256, 1992.

[17] Linder D. and Harden J., "An adaptive and fault tolerant wormhole routing strategy for *k*-ary hypecubes," *IEEE Trans. Computers* , vol. 40, no. 1, pp. 2-12, 1991.

[18] Ni L. M. and McKinley P. K., "A survey of routing techniques in wormhole networks," *IEEE Computer*, vol. 26, no. 2, pp. 62-76, 1993.

[19] Noakes M. *et al*, "The J-machine multicomputer: An architectural evaluation," *in Proceedings of 20th Int. Symp, Computer Architecture*, 1993.

[20] Nugent S. F., "The iPSC/2 Direct-Connect communication technology," *in Proceedings of Conf. on Hypercube Concurrent Computers & Applications*, vol. 1, pp. 51-60, 1988.

[21] Raghavendra C. S., Yang P. J., and Tien S. B., "Free dimension: an effective approach to achieving fault tolerance in hypercubes," *in Proceedings of 22nd Int. Symp. on Fault-Tolerant Comp.*, pp. 170- 177, 1992.

[22] Saad Y. and Schultz M. H., *Data communication in hypercubes*, Technical Report YALEU/DCS/RR-428, Computer Science Dept., Yale Univ., 1985.

[23] Sheu J. P. and Su M.Y., "A multicast algorithm for hypercube multiprocessors," *in Proceedings of Int. Conf. Parallel Processing*, pp. 18-22, 1992.

[24] Vanvoorst B., Seidel S., and Barscz E., "Workload of an iPSC/860," *in Proceedings of Scalable High-Performance Computing Conf.*, pp.221-228, 1994.

[25] Wu J., "Adaptive fault-tolerant routing in cube-based multicomputers using safety vectors," *IEEE Trans. Parallel and Distributed Systems*, vol. 9, no. 4, pp. 321-334, 1998.

[26] Wu J. and Fernandez E. B., "Broadcasting in faulty hypercubes," *in Proceedings of 11th Symp. Reliable Distributed Systems*, pp. 122-129, 1992.

Jehad Al-Sadi received his BSc degree in computer science from Tennessee State University, USA, in 1989, MSc degree in computer science from Jackson State University, USA, in 1993, and PhD degree in computer science from University of Glasgow, UK. He is currently an assistant professor in the Department of Computer Science at Zarka Private University, Jordan.

Khaled Day received an engineering degree in computer science from the University of Tunis in 1986 and the MSc and PhD degree from the University of Minnesota in 1989 and 1992, respectively. He is currently serving as associate professor and head of the Department of Computer Science of Sultan Qaboos University in Oman. His areas of interest include interconnection networks, parallel algorithms, distributed systems, and computer networks. He is a member of the IEEE.

Mohamed Ould-Khaoua received his BSc degree from the University of Algiers, Algeria, in 1986, and the MSc and PhD degrees in computer science from the University of Glasgow, UK, in 1990 and 1994, respectively. He is currently a lecturer in the Department of Computer Science at the University of Glasgow, UK. His research focuses on applying theoretical results from stochastic processor and queuing theory to the quantitative study of hardware and software architectures. His currently research interest are performance modeling/evaluation of parallel and distributed systems, parallel algorithms, ATM and multimedia networks. He is a member of the IEEE-CS, BCS, and C.Eng.