Discrete Applied Mathematics 37/38 (1992) 245-263
North-Holland

Analysis of minimal path routing schemes in the presence of faults

Jesse M. Gordon*

Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, MI 48109-2122, USA

Received 7 June 1989
Revised 25 March 1990

Abstract

Gordon, J.M., Analysis of minimal path routing schemes in the presence of faults, Discrete Applied Mathematics 37/38 (1992) 245-263.

The design and analysis of fault tolerant message routing schemes for large parallel systems has been the focus of much recent research. In this paper, we present a framework for the analysis of routing schemes in distributed memory multiprocessor systems containing faulty or unusable components. We introduce techniques for the derivation of the probabilities of successfully routing a single message using minimal path routing schemes. Using this framework, we derive closed form solutions for a wide range of routing schemes on the hypercube and on the two-dimensional mesh. The results obtained show the surprising resilience of the hypercube to a potentially large number of faults while demonstrating the inability of the mesh to tolerate a comparatively smaller number of faults.

As the size of parallel computer systems grows larger, so does the probability of component failure. Since the corresponding mean time to failure is likely to be short, off-line fault diagnosis and the subsequent replacement of failed units is not an attractive alternative for dealing with the problem. Rather, we would like to be able to continue system operation in the presence of such failures. A fundamental component of the operation of distributed memory multiprocessor systems is the routing of messages. The large size of these parallel systems mitigates against the selection of routing algorithms which require the propagation of *global* fault information to individual processors. Rather, we would prefer to use approaches which require only *local* fault information, such as the fault status of immediate neighbors. For similar reasons, we would prefer to use distributed (as opposed to centralized) routing schemes, i.e., ones in which routing decisions are made at the individual processors.

* Supported in part by NSF Grant No. DCR-85077851 and a fellowship from the Unisys Corporation.
Current address: IBM Corporation, 11400 Burnet Road, Zip 9641, Austin, TX 78758, USA.

Many researchers have studied the problem of developing fault tolerant routing algorithms. Recently, a number of researchers have studied the problem for systems in which massive numbers of faults are possible [1,9,16]. Under such conditions, schemes which assume local fault information and make routing decisions using local control are natural ones to study. In general, these papers have built on the fundamental work of Valiant [17], extending his paradigm to produce performance bounds for full permutation routing. In order to be able to provide these analytic bounds, the routing methods used are quite involved. We choose to attack the problem in a different manner, concentrating on deriving closed form solutions for the success probability of routing a *single* message using a variety of simple minimal path routing schemes. We show that the performance of such routing schemes, operating under simple assumptions, is already quite good. The performance bounds we obtain provide useful lower bounds for the performance of more complicated schemes. The analysis we perform also pinpoints the shortcomings of these schemes, identifying the areas upon which to focus to improve their fault tolerance.

We obtain the following results about the performance of simple single message minimal path routing schemes on hypercubes and two-dimensional meshes. For routing schemes which use no fault information, we derive closed form solutions for the success probabilities of routing the single message. We show that all these probabilities tend to 0 asymptotically. For routing schemes which use one step local information, we show that on a hypercube the probability of successfully routing a single message n steps is $\prod_{k=2}^{n}(1-p^k)$. On a mesh of dimension n, $(1-p)^{2n-3}\sum_{i=0}^{n-1}\binom{2n-1}{i}\times[1-2i/(2n-1)]p^i$ is derived as the probability of successfully routing a message the maximal distance ($2n-2$ steps) using the routing scheme with the best performance. These results confirm that hypercube schemes exhibit performance superior to the mesh schemes. For example, using the above closed forms, the asymptotic hypercube performance for $p=1/2$, or half of the nodes faulty, is a success probability of 57.8%. For a mesh of size 50 by 50, with p as low as 0.1, the success probability is only 7.7%. We also provide a total ordering of four different mesh routing schemes based on their performance as fault tolerant routing schemes.

Our results also have implications for certain connectivity properties of random hypercubes and meshes. The success probabilities obtained provide lower bounds for the probability that a randomly chosen pair of maximally distant nodes are in the same connected component. Similarly, they bound from below the probability that such a pair of nodes are connected by minimal length path consisting solely of nonfaulty nodes.

This paper is organized into four remaining sections. In Section 1, we present our framework for the analysis of minimal path fault tolerant routing schemes. Using this framework, in Sections 2 and 3 we derive techniques for analyzing the performance of routing schemes and apply them to obtain closed form solutions for routing on the hypercube and mesh. In Section 4, we summarize the ideas presented in the paper and list some resulting open problems and questions.

1. Preliminaries

We study two specific parallel systems in this paper: *hypercubes* and *two-dimensional meshes*. A hypercube contains $N=2^n$ processors addressed by n-bit strings, and any two processors are connected by a (bidirectional) communication link if their addresses differ in exactly one bit position. A two-dimensional mesh contains $N=n^2$ processors addressed by integer pairs (x, y) with $0\leq x, y\leq n-1$, arranged so that the processor with address (x_1, y_1) is connected to the processor labeled (x_2, y_2) if either $x_1=x_2$ and $|y_1-y_2|=1$ or if $|x_1-x_2|=1$ and $y_1=y_2$. These two systems can be viewed as special cases of a product graph known as the *k-ary n-cube*, in which processors are labeled by an n-tuple of integers in the range 0 to $k-1$. Two processors are connected if their labels differ by 1 in exactly one position and are the same in all other positions. Viewed in this manner the hypercube is a 2-ary (or binary) n-cube and the mesh is an n-ary 2-cube. Both of these systems are distributed memory, packet switched systems in which communication time is assumed to predominate, and local processing time can be ignored. In a packet switched system, messages are transmitted in units called *packets*; packets being routed from one processor to another are temporarily stored in the memory of intermediate processors and are later forwarded to their destinations. In this paper, we will concern ourselves solely with short messages, i.e., those which are no longer than one packet in length. For longer messages, it is easy to see that the probability that a long message is routed successfully is equal to the product of the probabilities of successfully routing each of its packets.

1.1. Fault distribution models

For the purposes of this study, a *fault* refers to a processor which is unavailable for use in communicating messages. This working definition is quite general and includes hardware failures as well as congested processors (i.e., communications hot spots). We assume that only nonfaulty, or *live*, processors take part in message communications. If a processor is faulty, this is the same as saying that all of its communication links are faulty too. Hence, the effect of a processor failure is at least as severe as the effect of a communication link failure, and is usually worse.

Two probability models commonly used in the study of random graphs [3,14] are particularly applicable to modeling permanent faults in multiprocessor systems.

(1) *Model A*. Fix p, the probability that a single node is faulty, $0\leq p\leq 1$, and let the set of node faults be distributed binomially with probability p.

(2) *Model B*. Fix f, the number of faults, $0\leq f\leq N$, and let all possible distributions of f faulty nodes be equally likely.

We assume that the distribution of faults is chosen before any routing occurs and that this fault pattern remains fixed for the duration of the routing attempt.

Results obtained using Model A and results obtained using Model B are closely related. In general, computations are easier to perform using Model A. When we

wish to compare results obtained for Model B with those obtained for Model A, we set $f = \lfloor pN \rfloor$.

Both of these models have been used extensively in the fault tolerance literature. For example, the papers [1,9,16] all analyze routing in situations where faults are distributed binomially, as in Model A. A short list of papers which use Model B to model the distribution of faults includes [4,5,6,10,15].

1.2. Routing information models

There are two *routing information models* which we use in our analysis to represent the amount of information available to individual nodes.

(1) *Model* 0. The *no local information* model. Individual nodes do not know the fault status of any other nodes in the system.

(2) *Model* 1. The *one-step local information* model. Individual nodes know which, if any, of their immediate neighbors are faulty.

1.3. Routing approaches

There are two basic approaches to message routing, *deterministic* and *random* schemes. In deterministic routing, the decision of which path the packet should take is fixed and predetermined given the packet source, destination and, possibly, the current packet location and the fault status of (neighboring) nodes encountered along the way. Alternately, in a random scheme, a random choice among all the possibilities determines the course of the routing. A useful way to view such probabilistic algorithms is given in [13]:

> Any probabilistic algorithm $\mathscr{A}$ can be transformed into an equivalent form in which the random source of $\mathscr{A}$ first chooses a deterministic algorithm from a predefined collection of deterministic algorithms $\mathscr{B} = \{\mathscr{B}_1, \mathscr{B}_2, \dots\}$ and then the selected algorithm is deterministically executed.

The focus of this paper is on the routing of a single message on a large multiprocessor system which may or may not contain faulty components. Hence, we restrict our attention to *local* routing schemes, i.e., those that make the decision of which processor to route to next based solely upon the current and destination processors. Our analysis is similarly restricted to *minimal path* schemes. In a minimal path scheme, each step traveled by a message decreases the distance between its current location and its destination.

In both hypercubes and meshes, local routing decisions are made based on the status of all neighbors which lie on minimal paths between the current location and the message destination. On the hypercube, the standard deterministic scheme is to choose to traverse the lowest numbered available dimension while a random scheme will route over a dimension selected uniformly at random from among the available

dimensions. An available dimension is one in which the value in that bit position is different in the labels of the current location and the destination. When the dimension is traversed, the differing (or wrong) bit position is said to have been corrected. These two approaches combined with the two routing information models yield four simple hypercube routing schemes to be analyzed. Table 1 summarizes the local decisions made by each of the four algorithms.

Since the mesh is not homogeneous, i.e., it does not look the same from every node, it is not the case that all minimal length paths are the same. Consider the local decision at a node which is a steps away from the destination in the x-direction and b steps away in the y-direction. Then there are two natural approaches to routing schemes.

(1) *Direction uniform*: We consider the local choice to be between the two directions in which we must route. For a deterministic scheme, then, we fix a favored direction, say the x-direction, and route in that direction if it is available. In a random scheme, we simply choose with equal probability between the two dimensions whenever there is a choice.

(2) *Path uniform*: If we treat the two directions as equals, then random routing produces a distribution which favors certain paths over others. In a path uniform scheme, we correct this problem by making our local decisions so as to leave the maximum number of possible remaining (minimal) paths. For a deterministic scheme, then, we route in the direction which has the most steps to go, choosing the favored x-direction whenever $a=b$. In a random scheme, we route in the x-direction with probability $a/(a+b)$ and in the y-direction with probability $b/(a+b)$.

An interesting way to view the two deterministic schemes is that the direction uniform scheme has a preferred *global* direction, whereas the path uniform scheme has a preferred *local* direction.

We have described four basic mesh routing schemes: *deterministic direction uniform*, *random direction uniform*, *deterministic path uniform* and *random path uniform*. Note that we can easily generalize the four approaches to apply to routing on a generalized mesh, such as the k-ary n-cube. For the hypercube, since $k=2$, the path and direction uniform schemes become one and the same.

Routing succeeds if the message reaches the destination. With no local information (Model 0), routing fails if we attempt to route the message to a faulty node. With one-step local information (Model 1), routing fails if we reach a *blocked* node, one which has no nonfaulty neighbors along any minimal path to the destination.

Table 1. Hypercube routing schemes

	Deterministic	Random
Model 0	Correct lowest dimension wrong bit	Correct a bit at random from the set of all wrong bits
Model 1	Correct lowest dimension *nonfaulty* wrong bit	Correct a bit at random from the set of all *nonfaulty* wrong bits

1.4. Assumptions

The problem we are concerned with is to analyze the probability of success of routing a single message on the two different architectures. Before any results are presented, we first summarize the assumptions under which we are working. First, we presume that the message to be routed is a single, short and indivisible message (that is, at most one transmission packet in length). Second, we assume that only node failures may occur and that the pattern of failures is fixed for the duration of the routing attempt. Third, we are operating under two different routing information models: Model 0, where individual nodes have no fault information, and Model 1, where individual nodes have local information of the fault status of each of their neighbors. Fourth, we are modeling the distribution of faults in two ways: Model A, where nodes fail independently with probability p, and Model B, where a fixed number, f, of nodes fails (with all possible distributions of f faults equally likely).

There are two additional assumptions which have important implications for the results we are about to present. First is the assumption that both the source and destination nodes are not faulty. This assumption is a reasonable one to make in the single message case. It does, however, affect the probabilities obtained for successful message routing. But, it is easy to see that any answers derived with this assumption can be converted to answers in the case where no assumption has been made. Specifically, if x is the probability of successful message routing from a live source to a live destination, then in Model A, $(1-p)^2x$ is the answer if nothing is known about the source or the destination; in Model B, the corresponding solution is $x\binom{N-2}{f-2}/\binom{N}{f} = (1-f/N)(1-(f-1)/N)x$. In particular, this assumption means that any results obtained with a high probability of node failure will be discounted heavily (as $(1-p)^2x$ will be a small percentage of the original answer).

The second integral assumption is that the source and destination nodes are maximally far apart (antipodal on a hypercube, opposite corners on a mesh). This assumption is made since our area of concern is minimal path algorithms. In that case, if the communicating nodes are not as far apart as possible, then we are really examining the problem on the subtopology delimited by those two nodes.

2. Routing with no information

Under the assumptions of Model 0, individual nodes have no information about the fault status of their neighbors. Our single message can be successfully routed only if there are *no faults* on the path chosen from the source to the destination. It is easy to see that, using Model 0, the probability of successful routing is the same using deterministic and random routing. This follows since once a random scheme chooses a path to route along, the message proceeds as if it were being routed deterministically. Since for each choice of path the probability of success is the same, it is also the same when we average out over all possible paths.

The argument presented above is independent of the topology of the underlying architecture; hence, we can present extremely general results for the Model 0 case. The key factor in determining the success of a routing scheme is simply the distance, d, between the source and destination nodes. The message will be routed successfully if and only if all of the nodes on the chosen path are live. If we denote the success probability using Model A as p_A^0 and that for Model B as p_B^0 (and recall that N stands for the total number of nodes in the system), then

$$p_A^0(N,d,p)=(1-p)^{d-1}$$

and

$$p_B^0(N,d,f)=\binom{N-2-(d-1)}{f}\Big/\binom{N-2}{f}.$$

Theorem 2.1. *Given a fixed interconnection network with N nodes and diameter d (a function of N) such that $d=o(N^{1/2})$, then for any $0\le p\le 1$, $\lim_{N\to\infty} p_B^0(N,d,\lfloor p(N-2)\rfloor)/p_A^0(N,d,p)=1$.*

Proof. By expanding out and canceling, we see that

$$p_B^0(N,d,f)=\prod_{i=0}^{d-2}\left(1-\frac{f}{N-2-i}\right).$$

Now, using the simple bounds that $x-1<\lfloor x\rfloor\le x$ and substituting in $\lfloor p(N-2)\rfloor$ for f, we get

$$\prod_{i=0}^{d-2}\left((1-p)+\frac{pi-1}{N-2-i}\right)<p_B^0(N,d,\lfloor p(N-2)\rfloor)\le\prod_{i=0}^{d-2}\left((1-p)+\frac{pi}{N-2-i}\right).$$

At this point, we divide through by $p_A^0(N,d,p)$, use the fact that $1+x\le e^x=\exp(x)$, and then apply crude bounds to both sides. This yields

$$1<p_B^0(N,d,\lfloor p(N-2)\rfloor)/p_A^0(N,d,p)\le\prod_{i=0}^{d-2}\left(1+\frac{pi}{(1-p)(N-2-i)}\right)$$

$$\le\exp\left(\sum_{i=0}^{d-2}\frac{pd}{(1-p)(N-d)}\right)\le\exp\left(\frac{p}{1-p}\,\frac{d^2}{N-d}\right).$$

But now since $d=o(N^{1/2})$,

$$\lim_{N\to\infty}\exp\left(\frac{p}{1-p}\,\frac{d^2}{N-d}\right)=\exp(0)=1$$

we have

$$1\le\lim_{N\to\infty}p_B^0(N,d,\lfloor p(N-2)\rfloor)/p_A^0(N,d,p)\le 1$$

which is the desired result. □

Results of this type are common in random graph theory. In this context, the theorem says that we can use whichever closed form probability is more convenient for our purposes. In fact, the proof shows that the ratio converges to 1 quite quickly; for the hypercube, the ratio converges on the order of $\exp(n^2/2^n)$, where n is the dimension of the hypercube. Unfortunately, this result does not apply to the (two-dimensional) mesh since the diameter is $2n-2$ while $N=n^2$; in this case, the ratio of the two success probabilities converges to some constant. However, this is not a great concern since both $\lim_{N\to\infty} p_A^0(N,d,p)=0$ for fixed $p>0$, and $\lim_{N\to\infty} p_B^0(N,d,f)=0$ for fixed $f/N>0$. This says that as system size grows large, all routing schemes which use no information have success probabilities which tend to 0 and, as such, are not good fault tolerant schemes.

3. Routing with one step local information

The routing schemes analyzed in this section use Model 1, where individual processors have complete knowledge of the fault status of each of their immediate neighbors. We introduce and apply two basic techniques for deriving success probabilities in this section. For a deterministic scheme, we observe that successful routing means that one of the possible minimal paths was traversed. The Model 0 solution tells us the probability that routing succeeds given the chosen path. To calculate the Model 1 solution, we must determine the number of faults required to make the chosen path the deterministic one. If we then sum over all possible paths, we have the desired solution. For the two architectures considered in this paper, this approach yields nice closed form solutions. For a random scheme, using Model A, we proceed from the source outward step by step. For each node at the current distance, d, the probability of routing to that node is the sum over all of its neighbors at distance $d-1$ of the probability of reaching that neighbor times the probability of choosing to go to the current node. We continue this calculation until we reach the destination node. For both the hypercube and the mesh, this approach also leads to closed form solutions.

3.1. Hypercube analyses

The first set of results to be presented are those for single message routing on a hypercube. Here, for the source and destination to be maximally far apart, one can choose any pair of antipodal nodes. We can do this since the cube is homogeneous.

We wish to analyze the probability of successfully routing a single message. Since there are two basic routing schemes and two fault distribution models, there are four separate analyses to perform. As is shown in the following section, by exploiting the relationship between deterministic and random routing in the single message case, we can reduce the number of analyses to be performed to two. We then derive closed form answers for the success probabilities in each of these cases.

3.1.1. The relationship between deterministic and random routing

Let us presume we are working with Model B, where for fixed cardinality, k, all subsets of faults of size k are equally likely to occur. Then, for single message routing on a hypercube, the probabilities of successful routing are *the same* using deterministic and random routing. We will show this to be true for Model B and then we will extend the results to cover Model A.

In order to present a clear proof, some notation is necessary. Let Q_n be the set of nodes of an n-cube and $\mathcal{Q}_n = 2^{Q_n}$ be its power set. Then a subset of faults (i.e., a subset of nodes of the hypercube designated as faulty) can be denoted as $\mathcal{F} \in \mathcal{Q}_n$. Now, if we fix a destination node $q_d \in Q_n$, we can define the a priori probability functions which map a source node and a fixed set of faults to the actual probability of successful message routing. More precisely, we let

- $\text{Det} : Q_n \times \mathcal{Q}_n \mapsto \{0, 1\}$ and
- $\text{Ran} : Q_n \times \mathcal{Q}_n \mapsto [0, 1]$

be the probability functions for deterministic or random routing using Model 1. Note that these functions are not probability measures since, in general, they do not sum to 1 over all possibilities. In addition, we define $\text{Det}(q_s, \mathcal{F}) = 0$ whenever $q_s \in \mathcal{F}$ or if $q_d \in \mathcal{F}$, and similarly for Ran. This definition simply says that routing to or from a faulty node never succeeds.

Next, we extend the notion of a fault set. The group of automorphisms of the cube which fix a vertex is S_n, the symmetric group of size n or the permutation group on n letters. We can see this by examining the fixed vertex. The automorphism leaves the cube looking the same from that vertex's point of view, so all it can have done is permuting its links. But this permutation of links is just a permutation of the cube's dimensions. We let the group S_n act on the set $\mathcal{Q}_n$. This partitions $\mathcal{Q}_n$ into orbits, or equivalence classes under the group action. Let us call any such orbit a *pattern*, denoted (in the standard fashion) $[\mathcal{F}]$ where $\mathcal{F}$ is the *representative* of the pattern. In other words, a pattern is a set of sets of faults which are invariant under any permutation of the dimensions of the cube. We can now extend the definitions of the probability functions above to maps from a node and a pattern, i.e., from a subset of $Q_n \times 2^{\mathcal{Q}_n}$, in the natural manner: $\text{Det}(q_s, [\mathcal{F}]) = \sum_{\mathcal{F}' \in [\mathcal{F}]} \text{Det}(q_s, \mathcal{F}')$.

We now state the theorem we wish to prove:

Theorem 3.1. *In Model B, for* $q_s \in Q_n$ *and* $0 \le f \le 2^n$, $\sum_{\mathcal{F} \in \mathcal{Q}_n, |\mathcal{F}| = f} \text{Det}(q_s, \mathcal{F}) = \sum_{\mathcal{F} \in \mathcal{Q}_n, |\mathcal{F}| = f} \text{Ran}(q_s, \mathcal{F})$.

In other words, the probabilities are the same for a random set of faults of fixed size f. The basic idea behind the proof is that the probabilities of success using all of the random paths average out so that their probability is the same as that for the deterministic path. Note that we are assuming the source and destination nodes referred to in this theorem and the subsequent proof are both live. This assumption is made for convenience as the other three cases are trivial.

The following notation will be used in the proof of the theorem. We let the Hamming distance between two nodes (number of steps, or differing bit positions) be denoted as a function $h: Q_n \times Q_n \mapsto \{0, 1, \ldots, n\}$, and then for a fixed destination node, $q_d \in Q_n$, use the shorthand $h(q_s)$ for $h(q_s, q_d)$. Lastly, we denote by $\langle q_s, q_d \rangle$ the subcube induced by nodes q_s and q_d, i.e., the set of all nodes on minimal paths between q_s and q_d.

The proof of the theorem depends upon the following key lemma. It shows that the sum of the success probabilities over all elements of a pattern is the same for deterministic and random routing.

Lemma 3.2. *In Model B, given any* $q_s \in Q_n$ *and any* $\mathscr{F} \in \mathscr{Q}_n$, $\mathrm{Det}(q_s, [\mathscr{F}]) = \mathrm{Ran}(q_s, [\mathscr{F}])$.

Proof. By induction on $h(q_s)$. The statement is clearly true for all nodes q with $h(q) = 1$ since both deterministic and random routing are the same from one step away. Now, assume the hypothesis is true for all nodes q with $h(q) < k$ and all choices of $\mathscr{F}$. Choose node q_s such that $h(q_s) = k$. Label the neighbors of q_s in $\langle q_s, q_d \rangle$ as $x_1, x_2, \ldots, x_k$ and assume that $j \leq k$ of them are live. Then,

$$\begin{aligned}
\mathrm{Det}(q_s, [\mathscr{F}]) &= \sum_{\mathscr{F}' \in [\mathscr{F}]} \mathrm{Det}(q_s, \mathscr{F}') \\
&= \sum_{i=1}^{k} \sum_{\substack{\mathscr{F}' \in [\mathscr{F}] \\ x_i \text{ first}}} \mathrm{Det}(x_i, \mathscr{F}') \\
&= \sum_{i=1}^{k} \frac{1}{j} \sum_{\mathscr{F}' \in [\mathscr{F}]} \mathrm{Det}(x_i, \mathscr{F}') \\
&= \sum_{i=1}^{k} \frac{1}{j} \mathrm{Det}(x_i, [\mathscr{F}]) \\
&= \sum_{i=1}^{k} \frac{1}{j} \mathrm{Ran}(x_i, [\mathscr{F}]) \\
&= \mathrm{Ran}(q_s, [\mathscr{F}]).
\end{aligned}$$

The first line is by definition. The next line expresses the fact that deterministic routing proceeds by sending the message to the first live neighbor. It also uses the fact that an automorphism of the cube which fixes q_s and q_d does not change the forward and backward neighbors of a particular node (q_s in this case), but merely their ordering. The next line is the key line in the proof. To see that it is true, first observe that over all automorphisms of the cube each live x_i is first exactly $1/j$ of the time. In fact, each live x_i appears in each of the j possible positions exactly $1/j$ of the time. But $\mathrm{Det}(x_i, \mathscr{F})$ is independent of the position of x_i with respect to q. Similarly, $\sum_{\mathscr{F}' \in [\mathscr{F}]} \mathrm{Det}(x_i, \mathscr{F}')$ is independent of the position of x_i with respect to q. So we can break down this last sum as the sum over those $\mathscr{F}$ which place x_i first,

plus the sum over those which place x_i second, all the way up the sum over those $\mathcal{F}$ which place x_i jth. And all of these sums over these subsets of $[\mathcal{F}]$ are exactly the same. Hence, we have overcounted the sum by j times and the key line holds. Another way to view this is to look only at $\langle x_i, q_d \rangle$. We are only interested in automorphisms of this subcube. But when we sum over all the automorphisms of $\langle q_s, q_d \rangle$, we will sum over the automorphisms of $\langle x_i, q_d \rangle$ j times. Each subsum must come to the same total. Now, the remainder of the proof is straightforward. We rewrite the key line (by definition), apply the induction hypothesis and proceed to the conclusion. The last line says that random routing proceeds by routing to a random live neighbor in the forward direction. □

Proof of Theorem 3.1. Fix f and q and let $\mathcal{F}_1, \ldots, \mathcal{F}_m$ be the representatives of all of the patterns of each $\mathcal{F} \in \mathcal{Q}_n$ such that $|\mathcal{F}| = f$. Then,

$$\sum_{\substack{\mathcal{F} \in \mathcal{Q}_n \\ |\mathcal{F}| = f}} \mathrm{Det}(q, \mathcal{F}) = \sum_{i=1}^{m} \mathrm{Det}(q, [\mathcal{F}_i]) = \sum_{i=1}^{m} \mathrm{Ran}(q, [\mathcal{F}_i]) = \sum_{\substack{\mathcal{F} \in \mathcal{Q}_n \\ |\mathcal{F}| = f}} \mathrm{Ran}(q, \mathcal{F}).$$

The middle equal sign is true by the lemma. □

Finally, even though this result was defined in the context of Model B, it is easy to see that an analogous result holds for Model A. This is true because all fault sets of fixed cardinality f are also equiprobable when faults are distributed binomially. Hence, we have shown that for both fault distribution models, the success probabilities for deterministic and random routing of a single message on a hypercube are the same.

3.1.2. Derivation of closed forms

To derive the desired closed form solutions, we apply the techniques described in the introduction to Section 3. For deterministic routing, we need to calculate for each minimal path the number of faults required to make that path the deterministic choice. In the hypercube, there are $n!$ possible minimal path deterministic routes, each corresponding to the order in which the n dimensions are traversed. Hence, routes correspond to permutations on n elements. At any given time in the routing, the deterministic choice is to traverse the lowest numbered remaining untraversed dimension. If instead we traverse, say, the ith lowest dimension, this indicates the presence of $i-1$ faults, since each neighbor in the first $i-1$ dimensions must be faulty for the ith neighbor to be chosen. This insight says that given a specific route σ, i.e., a permutation on $1, \ldots, n$, the number of faults required for σ to be the deterministic path is equal to the number of inversions of σ, which is $|\{(i,j): 1 \le i < j \le n,\ \sigma(i) > \sigma(j)\}|$.

Following Knuth [12], let $I_n(k)$ be the number of permutations on n elements with exactly k inversions. Note that $\binom{n}{2}$ is the maximum number of inversions for

any permutation on n elements. Denoting the desired probability as p_A^1, we have shown that:

$$p_A^1(n,p) = (1-p)^{n-1} \sum_{k=0}^{\binom{n}{2}} I_n(k) p^k.$$

Note that $\sum_{k=0}^{\binom{n}{2}} I_n(k)p^k$ is the generating function for inversions. By using an alternate form of this generating function [12], we can simplify this result as follows:

$$\begin{aligned} p_A^1(n,p) &= (1-p)^{n-1} \sum_{k=0}^{\binom{n}{2}} I_n(k) p^k \\ &= (1-p)^{n-1} \prod_{k=0}^{n-1} (1+p+p^2+\cdots+p^k) \\ &= \prod_{k=2}^{n} (1-p^k). \end{aligned}$$

This result can also be arrived at in an alternate fashion by applying the derivation technique for random routing schemes.

Proof: Consider the local routing decision of a node k steps away from the destination. The probability that all k neighbors in the forward direction are faulty is p^k, so the probability of successfully routing the message this one step is $(1-p^k)$. The overall probability of successfully routing the message is the probability that we can route the message each step from $k=n$ steps away down to $k=2$ steps away (since the destination is assumed to be live). Hence, $p_A^1(n,p) = \prod_{k=2}^{n} (1-p^k)$.

Now, from real analysis, we know that $\lim_{n\to\infty} p_A^1(n,p)$ converges. Of course, so did $p_A^0(2^n,n,p)$ but it converged to 0. Unfortunately, we cannot derive a simpler closed form for this value; however, we can compute and graph its values quite easily. Figure 1 shows the values for $p_A^1(n,p)$ as a function of p.

Using the same insights that gave us:

$$p_A^1(n,p) = (1-p)^{n-1} \sum_{k=0}^{\binom{n}{2}} I_n(k) p^k$$

we see that

$$p_B^1(n,f) = \frac{1}{\binom{2^n-2}{f}} \sum_{k=0}^{\binom{n}{2}} I_n(k) \binom{2^n-2-(n-1)-k}{f-k}.$$

In the Model B case, this says that for each path which requires k faults to be the deterministic one, we must then place the remaining $f-k$ faults not on the chosen path.

Theorem 3.3. *For any* $0 \le p \le 1$, $\lim_{n\to\infty} p_B^1(n, \lfloor p(2^n-2) \rfloor)/p_A^1(n,p) = 1$.

Proof. The proof proceeds in two parts. First, we extract out the parts of the closed forms that resemble the Model 0 solutions and derive similar bounds for their ratio. Then, we add back in the summation over inversions and derive the desired result.

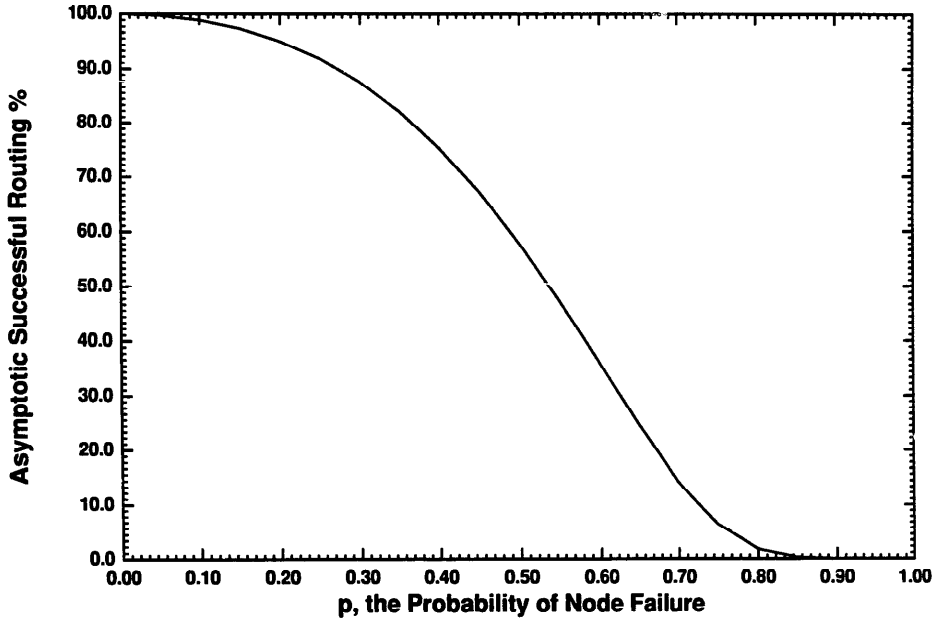

Fig. 1. $\lim_{n\to\infty} p_A^1(n,p)$: Hypercube random routing with one-step local information.

Let us set $r_A^k=(1-p)^{n-1}p^k$ and $r_B^k=\binom{2^n-2-(n-1)-k}{f}/\binom{2^n-2}{f}$. As in the Model 0 case, if we expand out r_B^k, apply simple bounds for the floor function, substituting in $\lfloor p(2^n-2)\rfloor$ for f, and then divide by r_A^k, we get

$$1<r_B^k/r_A^k\leq\exp\left(c\frac{n^5}{2^{2n}}\right)$$

for some constant c. Now we rewrite this equation as follows

$$r_A^k<r_B^k\leq r_A^k\exp\left(c\frac{n^5}{2^{2n}}\right).$$

Applying the summation bounds to all the parts of this equation yields

$$\sum_k I_n(k)r_A^k<\sum_k I_n(k)r_B^k\leq\sum_k I_n(k)r_A^k\exp\left(c\frac{n^5}{2^{2n}}\right).$$

But we can rewrite this more simply as

$$p_A^1(n,p)<p_B^1(n,\lfloor p(2^n-2)\rfloor)\leq\exp\left(c\frac{n^5}{2^{2n}}\right)p_A^1(n,p).$$

From here, we need just divide through by $p_A^1(n,p)$, and take the $\lim_{n\to\infty}$ and we have proven the theorem. □

As in the Model 0 case, the proof gives an estimate of the rate of convergence, which here is on the order of $\exp(n^5/2^{2n})$. We see that the ratio converges more

quickly in the Model 1 case than it does in the Model 0 case. Since we have now derived closed form solutions for each of the four possible model combinations, we have concluded our analysis of the performance of minimal path single message routing schemes on the hypercube.

3.2. *Mesh analyses*

The second architecture for which we analyze routing schemes is the two-dimensional mesh. The mesh is a popular architecture, especially for work in parallel image processing. If we use a model where the time to route a packet is proportional to the length of the wire it is being routed over, then the mesh can be shown to be an optimal architecture. If, instead, we use our standard model where packet routing times are equal and independent of wire length, then the bounded connectivity of the mesh dooms all routing schemes to poor performance. Since the connectivity is independent of the mesh size, we gain no benefits from increased mesh size; in fact, we show below that the performance of routing schemes is demonstrably worse for larger size meshes. We denote the probabilities we calculate by DD for deterministic direction uniform routing, RD for random direction uniform routing, and via DP and RP for the path uniform schemes.

3.2.1. *Direction uniform schemes*

We first consider the direction uniform case. For deterministic direction uniform routing, we reason much as we did for the hypercube deterministic case. Since the x-direction is the favored one, each time we route in the y-direction while the x-direction is also a choice represents a faulty node in the x-direction. Hence, the number of faults required for a given path to be the deterministic one is the number of y steps which precede the last x step. It is a simple matter to partition the possible paths by the number of y's they have before the last x, arriving at the following formulas.

$$\mathrm{DD_A}(n,p) = (1-p)^{2n-3} \sum_{i=0}^{n-1} \binom{n-2+i}{i} p^i$$

and

$$\mathrm{DD_B}(n.f) = \frac{1}{\binom{n^2-2}{f}} \sum_{i=0}^{n-1} \binom{n-2+i}{i} \binom{(n-1)^2-i}{f-i}.$$

The Model B solution merely counts, for each group of paths, the number of ways to place the remaining faults.

For random direction uniform routing, we first note that its success probabilities are not the same as those for deterministic direction uniform routing. This is one way in which the mesh is distinguished from the hypercube. To arrive at a closed form solution, we note that until all of the steps in one of the directions are exhausted, we have a possibility of two neighbors to choose from. In Model A, the probability that we can route the next step, with two neighbors, is $1-p^2$, or the

probability that we are not blocked. The probability of going to a specific one of the two neighbors is then $(1-p^2)/2$. If we have only one neighbor, the probability of routing the next step is $1-p$. Now, as in the deterministic case, we simply partition paths by the number of steps for which there are possibly two decisions to make. This is exactly the same as counting the number of y's before the last x, and vice versa. Since the mesh is symmetric in this respect, we will count paths which run out of x's first and double the answer. The closed form derived via this discussion is then

$$\mathrm{RD_A}(n,p) = 2\left(\frac{1-p^2}{2}\right)^{n-1}\sum_{i=0}^{n-2}\binom{n-2+i}{i}\left(\frac{1-p^2}{2}\right)^{i}(1-p)^{n-2-i}.$$

This formula says that we always have at least $n-1$ nodes with two forward neighbors; after that we count the number of different paths which have i additional such nodes. A comparable closed form solution for $\mathrm{RD_B}(n,f)$ has not been derived.

3.2.2. Path uniform schemes

For the deterministic path uniform scheme, the fixed path taken in the absence of faults strictly alternates steps in the x- and y-directions until reaching the destination. For each other path, the number of faults required to make it the deterministic one gives some measure of how far we have strayed from the diagonal. To do the counting in a systematic fashion, think of each local routing decision as having two outcomes: right (R, go in the direction with more to go) or wrong (W). Then a path from the source in one corner to the destination in the opposite corner corresponds to a string of length $2n-2$ consisting of R's and W's. The following lemma tells us which of the 2^{2n-2} possible strings correspond to deterministic path uniform routes.

Lemma 3.4. *Given a string of length* $2n-2$ *consisting of* R*'s and* W*'s. If, starting from the end of the string and working forward, the number of* R*'s always exceeds the number of* W*'s, then the string is valid (corresponds to a possible path); else the string is invalid.*

Proof. The basic idea is that each wrong requires one right to correct it. So, for a string to be valid, we must always have at least as many R's as W's in the remainder of the string (starting from the front, that is). But this is exactly the condition that the number of R's always exceeds the number of W's starting from the end of the string. For the invalid case, assume that the number of W's does exceed the number of R's and identify the point at which this first occurs. If there is a W in the last position, this is clearly impossible. If not, remove the W which must be the first element in the substring from the identified point to the end of the entire string. We are now left with a string in which the number of R's always exceeds the number of W's. This string must have an equal number of R's and W's, say k of each.

By assumption, we know this string ends in an R. But at best we have $k+1$ steps in the x-direction and the remaining $k-1$ steps in the y-direction. This is so since the step preceding this substring was a W. So the first $k-1$ wrongs must cause us to run out of one direction to go in. The first W after this occurs forces the string to be invalid. □

Now, we simply need to count the number of valid strings containing i W's. Since each W corresponds to a fault, we will have counted the number of paths made into the chosen path by the placement of exactly i faults. This problem has been studied in the literature (see Problem 2.2.1-4 of [11] or Section 5.1.4 of [12]). Hence, we have derived the following results:

$$\mathrm{DP_A}(n,p)=(1-p)^{2n-3}\sum_{i=0}^{n-1}\binom{2n-1}{i}\left[1-\frac{2i}{2n-1}\right]p^i$$

$$=(1-p)^{2n-3}\sum_{i=0}^{n-1}\left(\binom{2n-2}{i}-\binom{2n-2}{i-1}\right)p^i$$

and

$$\mathrm{DP_B}(n,f)=\frac{1}{\binom{n^2-2}{f}}\sum_{i=0}^{n-1}\binom{2n-1}{i}\left[1-\frac{2i}{2n-1}\right]\binom{(n-1)^2-i}{f-i}.$$

For the random path uniform scheme, we have been unable to directly compute a closed form solution for $\mathrm{RP_A}(n,p)$. We have used the random proof technique to calculate symbolically the exact solutions for small values of n. From examination of these values, we have observed that the polynomial (in p) is of degree $4n-6$, begins with $1-((n-2)/n)p$, and that after the constant term, the remaining terms alternate signs in pairs (i.e., two negative terms, then two positive terms, and so on). Derivation of a closed form for this scheme remains an open problem.

3.2.3. Asymptotics

Now that we have derived closed form solutions for the mesh routing schemes, we should like to see how their performance changes as mesh size grows larger. Unfortunately, the performance of these schemes all deteriorate asymptotically. This result is summarized in the following theorem.

Theorem 3.5. *Any mesh routing scheme R_A which uses one step local information and local control to make routing decisions has the property that* $\lim_{n\to\infty} R_A(n,p)=0$ *for fixed* $0<p\leq 1$.

Proof. At each step, the probability of being able to route one more step is no greater than $1-p^2$, the probability that both of the neighbors are not faulty. (If there is only one neighbor, observe that $1-p\leq 1-p^2$.) The total distance to traverse is $2n-2$ steps, so the probability of success is bounded from above by $(1-p^2)^{2n-2}$. Hence, $0<R_A(n,p)<(1-p^2)^{2n-2}$, and if we take the $\lim_{n\to\infty}$ we have proved the result. □

Despite the fact that all of the success probabilities vanish as mesh size increases, the closed forms obtained allow us to rank the four routing schemes in terms of their performance.

Conjecture 3.6. *For fixed* n *and* $0 \le p \le 1$, $DP_A(n,p) \ge RP_A(n,p) \ge RD_A(n,p) \ge DD_A(n,p)$.

The relationships between DP and RP, DP and RD, and RD and DD can be shown using the closed forms derived. A more useful way to look at this ordering is to understand the intuition behind the relationships. For the direction uniform schemes, the random scheme provides some chance for staying off of the edge of the mesh and, hence, yields performance no worse than the deterministic scheme. The path uniform schemes are superior to the direction uniform schemes since they leave open a greater number of possible paths at each decision point, therefore requiring more faults to sabotage. For the path uniform schemes, the deterministic scheme always goes in the direction of the largest number of remaining paths. Since the random scheme may choose to go in the opposite direction, its performance can be no better than that of the deterministic scheme.

A recent paper [2] analyzes deterministic path uniform routing in a related setting. They call their routing scheme the "Zig-Zag Shortest-Path Routing Policy" and use an inductive proof technique to show its optimality with respect to "maximizing the probability of reaching the destination from a given source without delays at intermediate nodes". The model they use is one where forward neighbors are polled in a specific order to test whether they are currently available to receive a message. We believe that the inductive proof technique used could be modified to help prove the conjecture discussed above.

Lastly, we note that even though the deterministic and random schemes have different success probabilities for the mesh, we can create a scheme which has exactly the same performance as the deterministic path uniform scheme. The new routing scheme simply goes in the direction with more steps remaining and chooses randomly between the two directions when there are equal number of steps remaining to be routed in each. The proof is much like the one which shows the equivalence between deterministic and random routing on a hypercube. Such a hybrid scheme provides the performance of the best scheme while adding the least amount of randomness to the algorithm.

4. Summary

We have presented a framework for the analysis of minimal path fault tolerant single message routing schemes on large parallel systems. We have introduced techniques for the derivation of success probabilities for routing that single message. By applying these techniques, we have derived closed form solutions for routing

schemes on the hypercube and on the two-dimensional mesh. The results obtained show that routing schemes which do not use local information provide little, if any, tolerance to faults. For routing using one-step local information, hypercube schemes were shown to be superior (at least asymptotically) to mesh schemes. In particular, the asymptotic performance of the hypercube schemes analyzed all converge to some constant value greater than zero; for meshes, the asymptotic performances all converge to 0. A complete ordering of mesh schemes by performance was conjectured, with the deterministic path uniform scheme achieving the best performance.

There are a number of areas for further research. One open problem is to derive a closed form solution for the random path uniform mesh scheme. Secondly, one might wish to derive proof techniques and analyses under the assumption that complete k-step fault information is available to each processor. Lastly, we mention two related problems which we have addressed in [7,8]. In [7], we applied the techniques presented in this paper and derived closed form solutions for success probabilities of routing schemes on the k-ary n-cube. In [8], we examined the problem of analyzing simple *non*-minimal path routing schemes.

Acknowledgement

The work in this paper could not have been completed without the advice, guidance and support of Quentin F. Stout. The author would also like to thank Kevin Compton, Deepak Sherlekar and Bruce Wagar for many fruitful discussions and useful suggestions. Finally, the comments and suggestions of the anonymous referees helped improve the presentation of this paper.

References

[1] F. Annexstein, Fault tolerance of hypercube-derivative networks, in: Proceedings of the 1989 ACM Symposium on Parallel Algorithms and Architectures (1989) 179–188.

[2] H.G. Badr and S. Podar, An optimal shortest-path routing policy for network computers with regular mesh-connected topologies, IEEE Trans. Comput. 38 (1989) 632–636.

[3] B. Bollobás, Random Graphs (Academic Press, London, 1985).

[4] A. Broder, D. Dolev, M. Fischer and B. Simons, Efficient fault tolerant routings in networks, in: Proceedings of the 16th Annual ACM Symposium on Theory of Computing (1984) 536–541.

[5] D. Dolev, J.Y. Halpern, B. Simons and H.R. Strong, A new look at fault-tolerant network routing, Inform. and Comput. 72 (1987) 180–196.

[6] P. Feldman, Fault tolerance of minimal path routings in a network, in: Proceedings of the 17th Annual ACM Symposium on Theory of Computing (1985) 327–334.

[7] J.M. Gordon, Efficient schemes for massively fault tolerant parallel communication, PhD thesis, The University of Michigan, Ann Arbor, MI (1990).

[8] J.M. Gordon and Q.F. Stout, Hypercube message routing in the presence of faults, in: Proceedings of the 3rd Conference on Hypercube Concurrent Computers and Applications (1988) 318–327.

[9] J. Hastad, T. Leighton and M. Newman, Fast computation using faulty hypercubes, in: Proceedings of the 21st Annual ACM Symposium on Theory of Computing (1989) 251–263.

[10] J.P. Hayes, A graph model for fault-tolerant computing systems, IEEE Trans. Comput. 25 (1976) 875–884.

[11] D.E. Knuth, The Art of Computer Programming, Vol. 1: Fundamental Algorithms (Addison-Wesley, Reading, MA, 1968).

[12] D.E. Knuth, The Art of Computer Programming, Vol. 3: Sorting and Searching (Addison-Wesley, Reading, MA, 1973).

[13] D. Krizanc, D. Peleg and E. Upfal, A time-randomness tradeoff for oblivious routing, in: Proceedings of the 20th Annual ACM Symposium of Theory of Computing (1988) 93–102.

[14] E.M. Palmer, Graphical Evolution: An Introduction to the Theory of Random Graphs (Wiley, New York, 1985).

[15] D. Peleg and B. Simons, On fault tolerant routings in general networks, Inform. and Comput. 74 (1987) 33–49.

[16] P. Raghavan, Robust algorithms for packet routing in a mesh, in: Proceedings of the 1989 ACM Symposium on Parallel Algorithms and Architectures (1988) 344–350.

[17] L.G. Valiant, A scheme for fast parallel communication, SIAM J. Comput. 11 (1982) 350–361.