Correspondence

Fault tolerant permutation mapping in multistage interconnection network

U. Maulik [a], S. Bandyopadhyay [b,*], S. Bhattacharyya [c]

[a] *Kalyani Engineering College, Kalyani, India*
[b] *Indian Statistical Institute, 203 Barrackpore Trunk Road, Calcutta 700 0035, India*
[c] *Jadavpur University, Calcutta, India*

Received 9 April 1997; received in revised form 15 October 1998; accepted 3 December 1998

Abstract

An efficient scheme for fault tolerant mapping of permutations is designed. The proposed algorithm uses extra passes through the network, instead of additional hardware.

Keywords: Fault tolerant routing; Interconnection network; Permutation; Stuck-at fault

1. Introduction

Permutation in multistage interconnection network (MIN) [1] is defined as a one to one exhaustive mapping from the set of inputs to the set of outputs. Presence of stuck-at faults in the switching elements of a MIN results in degradation of its permutation mapping capability. Research in this area has till now dealt with fault tolerant message routing from single source to single destination [2,3]. On the contrary, in this article we propose an algorithm for routing of permutations in presence of stuck-at faults by performing multiple passes. For generalized faults, a combination of the method proposed in this article and the one developed in [2] may be adopted as long as the network possesses dynamic full access capability (DFA).

2. Permutation mapping in faulty networks

A switch, represented by an ordered pair $\langle r, s\rangle$ where r and s represent the stage and the switch

Fig. 1. Stuck-at-faults.

number, respectively, can be kept in either straight (T) or exchange (X) configuration. A faulty switch is assumed to be stuck at any one of these two configurations (see Fig. 1). Properties of the MIN utilized for designing the algorithm are as follows:

Property 1 (Buddy property). *A pair of switches in a stage are called output buddies if the outputs of both these switches go to the same pair of switches in the next stage. Again, a pair of switches in a stage are called input buddies if the inputs of both these switches come from the same pair of switches in the previous stage.*

Property 2. *It can be easily proved that under fault free condition, a submapping from some input processor i_1 to an output processor θ_1 can always be achieved in two passes. In the first pass, all the switches in stage 0 to k are kept in their required configuration, i.e., those required to map i_1 to θ_1, while the remaining switches in stage $(k+1)$ to the last one along the route are set to T. In the second pass, the switches in stage 0 through k are kept in T while the rest are kept in their required configuration.*

2.1. In presence of stuck-at faults

Let *Fault set* F be the set of stuck-at faulty switches in the network. Multiple stuck-at faults in the network can be treated as a set of disjoint single faults if (i) F does not contain any switch from stage 0, (ii) for each of the switching elements in F, its input buddy is not in F and (iii) for each pair of switching elements S_i and S_j in F located in stages i and j, respectively, such that $i < j$, neither S_i nor its input buddy is an element of the input tree of S_j, where input tree is the binary tree generated on the input side taking S_j as the root.

Algorithm for fault tolerant routing: Given a network and its fault set F, let l_f_s be the last stage in which a fault occurs. In Pass 1, for each switch in F, its input buddy, *ib*, is set to X and the corresponding output buddies in the previous stage, ob_1 and ob_2, are set to a configuration opposite to one required for mapping π (or, *opp config*). All the other switches from stage 0 to $l\_f\_s - 1$ are set to the configurations required for mapping π (or, *reqd config*), while those in the stages l_f_s to the last stage are set to T.

In Pass 2, for each faulty switch in F, ob_1 and ob_2 are set to X. If the faulty switch is stuck-at-X, then *ib* is set to *opp config*, otherwise *ib* is set to *reqd config*. All the other switches from stage 0 to $l\_f\_s - 1$ are set to T, while those in the stages l_f_s to the last stage are set to *reqd config*.

Algorithm Fault-tolerant-routing( ):

Pass 1: For each switch in F set its *ib* to X and ob_1 & ob_2 to *opp config*.
Set other switches from 0 to $(l\_f\_s - 1)$ to the *reqd config*.
Set other switches from l_f_s to the last stage to T.

Pass 2: For each switch in F do
If (stuck-at T) then set its *ib* to *reqd config* else set its *ib* to *opp config*.
Set its ob_1 and ob_2 to X.

Set other switches from 0 to (l_f_s-1) to T.
Set other switches from l_f_s to the last stage to *reqd config*.

Example. Consider a faulty 8×8 Omega network shown in Fig. 2. The stuck-at T and stuck-at X faults in stage 1 (switches 1 and 2) are indicated by '**.T**' and '**.X**', respectively. Note that the input buddies of $\langle 1,1\rangle$ and $\langle 1,2\rangle$ are $\langle 1,0\rangle$ and $\langle 1,3\rangle$, respectively and the corresponding output buddies are $\langle 0,0\rangle$ and $\langle 0,2\rangle$ and $\langle 0,1\rangle$ and $\langle 0,3\rangle$, respectively. The settings shown inside each switch are those required for mapping

$$\pi = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 3 & 5 & 4 & 2 & 7 & 0 & 1 & 6 \end{pmatrix}.$$

The entries in the form 'x/y', $x, y \in \{X, T\}$, shown below each switch indicate that the corresponding switch is set to x in Pass 1 and to y in Pass 2. The permutations realized in the two passes are

$$\pi_1 = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 4 & 3 & 0 & 5 & 2 & 7 & 6 & 1 \end{pmatrix}$$

and

$$\pi_2 = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 4 & 6 & 7 & 5 & 3 & 2 & 1 & 0 \end{pmatrix},$$

respectively, thereby effectively realizing π.

2.2. In presence of generalized faults

Here, we generalize the results obtained in the previous section to include the case of unusable switches. For a single pass mappable $N \times N$ permutation, the N routes $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_N\}$ may be divided into three pairwise disjoint classes $\Sigma_1 = \{\sigma_{i_1}, \sigma_{i_2}, \ldots, \sigma_{i_M}\}$, $\Sigma_2 = \{\sigma_{j_1}, \sigma_{j_2}, \ldots, \sigma_{j_O}\}$, and $\Sigma_3 = \{\sigma_{k_1}, \sigma_{k_2}, \ldots, \sigma_{k_{N-M-O}}\}$, such that $\Sigma_1 \cup \Sigma_2 \cup \Sigma_3 = \Sigma$. Here Σ_1 are the paths unaffected by the faults, Σ_2 are the paths affected by stuck-at faults and Σ_3 are the paths that are blocked by the unusable faulty switches. Note that the paths of Σ_1

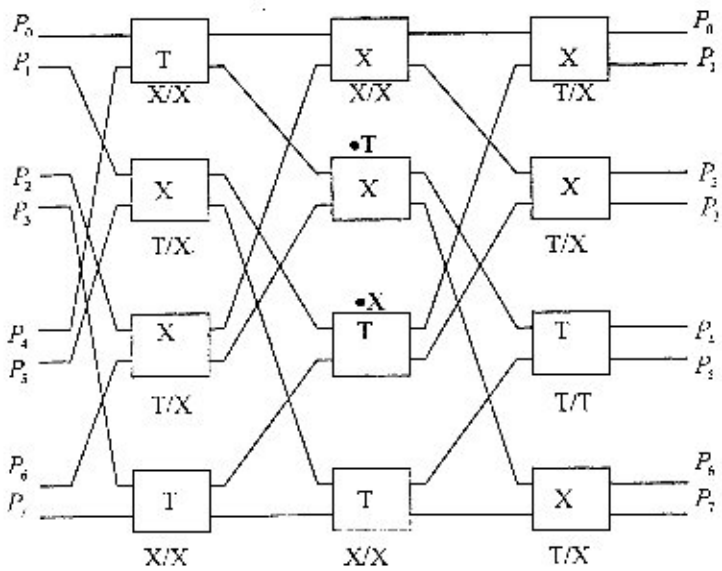


Fig. 2. Setting with in faulty 8×8 omega network.

can be mapped in a single pass and the paths of Σ_2 can be mapped in two passes using the algorithm described here. Paths of Σ_3 can be mapped utilizing the methodology described in [2], as long as the MIN possesses DFA property.

Consider Fig. 2 again for illustration. Let $\langle 1,1\rangle$ be stuck-at T and $\langle 1,2\rangle$ be unusable. As earlier, the permutation to be mapped is

$$\pi = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 3 & 5 & 4 & 2 & 7 & 0 & 1 & 6 \end{pmatrix}.$$

Here

$$\Sigma_1 \equiv \begin{pmatrix} 0 & 1 & 6 & 7 \\ 3 & 5 & 1 & 6 \end{pmatrix}, \quad \Sigma_2 \equiv \begin{pmatrix} 2 & 4 \\ 4 & 7 \end{pmatrix}, \quad \text{and}$$

$$\Sigma_3 \equiv \begin{pmatrix} 3 & 5 \\ 2 & 0 \end{pmatrix}.$$

Obviously, Σ_1 can be mapped in a single pass and Σ_2 can be mapped in two passes. Each path in Σ_3, $3 \to 2$ and $5 \to 0$, may be realized in two passes as $3 \to 6 \to 2$ and $5 \to 6 \to 0$ respectively. Note that none of these routes pass through the faulty switches.

3. Conclusion

The scheme for fault tolerant mapping of permutations designed in this article can be used for any MIN possessing buddy property as long as the network retains DFA property. The routing algorithm designed here can also be easily extended to map multiple pass permutations.

References

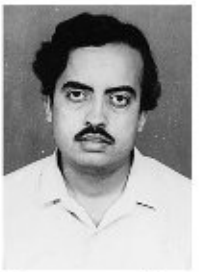
[1] K. Hwang, Advanced Computer Architecture: Parallelism, Scalability, Programmability. Computer Science Series, McGraw-Hill, New York, 1993.

[2] A. Verma, C.S. Raghavendra, Fault-tolerant routing in multistage interconnection networks, IEEE Trans. Comput. 38 (1989) 385–393.

[3] H.W. Chang, K.L. Chung, Fault-tolerant routing in unique-path multistage omega network, Information Processing Letters 44 (1992) 201–204.

Ujjwal Maulik obtained his Bachelor's degrees in Physics and Computer Science from Culcutta University, India, in 1986 and 1989 respectively. He did his Master's and Ph.D. degrees from Jadavpur University, India in 1991 and 1997 respectively. He is an Assistant Professor and Head of the department of Computer Science in Kalyani Engineering College, India, where he is currently also the office-incharge. He visited Los Alamos, USA, in 1997 as a scientist. His current research interests include parallel processing, interconnection networks, image processing and genetic algorithms.

Sanghamitra Bandyopadhyay did her Bachelor's degrees in Physics and Computer Science from Culcutta University, India, in 1988 and 1991 respectively. She received the A.K. Chowdhury Memorial Award for being adjudged the best student in Computer Science in 1991. She did her Master's degree in Computer Science from Indian Institute of Technology, Kharagpur, India, in 1993, where she was the first recipient of the Dr. Shankar Dayal Sharma Gold Medal for being adjudged the best all round post-graduate performer of 1992–1993, and the Institute Silver medal. Subsequently she obtained her Ph.D. degree from Indian Statistical Institute, Calcutta in 1998. She visited Los Alamos National Laboratory, New Mexico, USA in 1997 as a graduate research assistant. Her current research interests include interconnection networks, parallel processing, genetic algorithms, pattern recognition, neural networks and image processing.



Swapan Bhattacharya received his M.Tech. and Ph.D. degrees in Computer Science from University of Calcutta. India in 1981 and 1991 respectively. He also received his M.B.A. in Operations Research from Jadavpur University, India in 1987. A recipient of the Young Scientist Award from UNESCO in 1989, he has the experience of working in the faculty of computer science in various Universities around the world. His present permanent position is Professor in the Department of Computer Science and Engineering, Jadavpur University, India and currently he is visiting the Department of Computer Engineering, Atilim University, Ankara, Turkey. His research interests include distributed systems and software engineering. He may be reached at the e-mail address bswapan@hotmail.com.