Journal of Computer Science 4 (3): 220-224, 2008
ISSN 1549-3636
© 2008 Science Publications

Irregular Class of Multistage Interconnection Network in Parallel Processing

[1]Sandeep Sharma, [1]K.S. Kahlon, [2]P.K. Bansal and [3]Kawaljeet Singh
[1]Department of Computer Science and Engineering,
Guru Nanak Dev University, Amritsar, 143001, India
[2]MIMIT College of Engineering and Technology, Malout, India
[3]Computer Centre, Punjabi University, Patiala, India

Abstract: A major problem in designing a large-scale parallel and distributed system was the construction of an Interconnection Network (IN) to provide inter-processor communication. One of the biggest issues in the development of such a system was the development of an effective architecture and algorithms that have high reliability, give good performance (even in the presence of faults), low cost, low average path length, higher number of passes of request and a simple control. In this study, a new class of Irregular Fault Tolerant Multistage Interconnection Network (MIN) called Improved Four Tree (IFT) is introduced. Algorithms for computing the cost and permutations passable in the presence and absence of fault are developed for the analysis of various networks with proposed network.

Key words: Fault tolerant MIN, four tree network, multistage interconnection network, routing, permutation

INTRODUCTION

A Network is an interconnected collection of autonomous computers. Two computers are said to be interconnected if they are able to exchange information. The connection can make up through links [7]. A typical IN consists of a number of switching elements (SE's)[1,9] and interconnection links that enables the processor to communicate themselves or with memory units. As a compromise between two extremes (time-shared and crossbar networks) and various operation characteristics of an IN, multistage interconnecting network[2] was introduced. A multistage Interconnection network is capable of connecting an arbitrary input terminal to an arbitrary output terminal[7]. Generally a MIN consists of more than one stage of small interconnection networks called Switching Elements (SEs). An irregular class of multistage interconnection network for parallel processing named Improved Four Tree (IFT) has been proposed and analyzed in this study.

CONSTRUCTION METHOD OF IFT

A typical IFT is an Irregular Multistage Interconnection Network of size 2^n x 2^n is constructed with the help of two similar groups; lower and upper, each group consisting of sub network of $2^{n-1}\times 2^{n-1}$ size. It has $\log_2 N$-1 stages, both stages at $\log_2 N$-3 and $\log_2 N$-1 have 2^{n-1} switches (where N = 2^n of N x N network). The centre stage has exactly 2^{n-2} switches. The switches in a stage having same number from both the groups form a loop. Each source is connected to two different switches in each group with the help of multiplexer and each destination is connected with demultiplexer. In case the main route is busy or faulty, requests will be routed through alternate path in the sub-network. The advantage of this network is that if both switches in a loop are simultaneously faulty in any stage then even some sources are connected to the destinations which are not possible in Four Tree (FT)[4,6] Network. IFT network of size 16×16 is shown in Fig. 1.

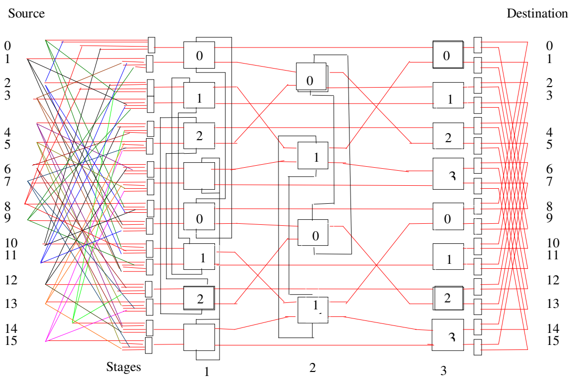

Fig. 1: IFT MIN of size N = 16

COST-EFFECTIVENESS ALGORITHM

To estimate the cost of a network, one common method is to calculate the switch complexity [3] with the assumption that the cost of a switch is proportional to the number of gates involved, which is roughly proportional to the number of 'cross points' within a switch [9]. So in this way the cost of each 2×2 switch has 4 units cost and each 3×3 switch has 9 unit cost and so on. For the interconnection network that contains multiplexers and demultiplexers, we roughly assume that each Mx1 multiplexers or 1×M demultiplexers has M units cost [5]. The cost effectiveness algorithm is as follows:

```
begin
    let cost = 0 ( Initialize a cost parameter)
    input the total no of switches say n
    input the total no of mux or demux say m
if n = = 0 and m = = 0 then
     cost = 0
      exit
else
    repeat
      begin
         enter the no of inputs of switch say t (switch
inputs may vary)
         cost = cost+ $t^t$
      n = n-1
until n>0
repeat
enter the no of inputs or outputs of mux or demux say t
cost = cost + t
m = m-1
Until m>0
```

PERMUTATION PASSABLE ALOGOTIHM

A one to one correspondence between source to destination is called Permutation [6]. One of the major aspects of permutation parameter is that it varies with path length. To find out this parameter for a network, it is assumed that source and destination is represented by:

S_i (where i = 0,1…N-1)
D_i (where i = 0,1…N-1)

Permutation can be evaluated in two ways:

Identity permutation: A one-to-one correspondence between same source and destination number is called Identity Permutation. In terms of source and destination this can be expressed by:
$S_i = D_i$
For I = 0,1…N-1

For example: Connectivity between source to destination for Identity is represented by:

Source: (0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15)
Destination: (0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15)

Incremental permutation: Incremental means that each source is connected to the destination in a circular chain and is represented as:

Source: (0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15)
Destination: (4 5 6 7 8 9 10 11 12 13 14 15 0 1 2 3)

We are considering the best possible cases to find out the permutations:

- If a fault is present in a single switch (Non-Critical Case)
- If the switches are faulty in a loop (Critical Case) if it exist

To find the permutation passable between a given source to destination, we need following parameters:

- Path Length of given source to destination
- Routing tag

Path length algorithm: There are always multiple paths of different path length exist for a given source to destination pair in fault tolerant network. The algorithm [4] for the allocation of path length gives the information about different possible paths between source and destination pair. The possible path length is varies from 2 to 2x-1 for a $2^n \times 2^n$ network, depending upon source to destination pair.
Let the source S and destination D be represented in binary code as:

$S = S_{n-1}\ldots\ldots\ldots S_1. S_0$
$D = D_{n-1}\ldots\ldots\ldots D_1. D_0$

The algorithm is as follows:

If $[(S_{n-2} + D_{n-2}) + (S_{n-3} + D_{n-3}) + \ldots. (S_1 + D_1)]$
Is Zero

Then minimum path length is 2 and all paths of different lengths are possible i.e., 4,6, (2x-2), (2x-1).

```
Else
If
If [( $S_{n-2} + D_{n-2}$ ) + ( $S_{n-3} + D_{n-3}$ ) + …. ( $S_2 + D_2$)]
 is zero then
all paths of length equal or greater than 4 are possible

else
:
:
if
If [( $S_{n-2} + D_{n-2}$ ) + ( $S_{n-3} + D_{n-3}$ ) + …. ( $S_p + D_p$)]
 is zero
{where 1< = p< = (n-2)}
then
all paths of length equal to or greater than 2p are possible
else
path of length 2x-1 (i.e longest path) is possible only.
```

Routing tag algorithm: The routing tag algorithm[4,8] for a given network gives the information between source and destination terminal pair for a given path length (if it exists).

```
If 2< = p<(2x-1)
then
routing tag = $S_{n-1}$ . $(1.1..1)_{([p/2]-1)}$ .0.( $d_{([p/2]-1}$ … $d_0$). $d_{n-1}$
 else
 if
 p = (2x-1)
then
routing tag = $S_{n-1}$ . $(1.1..1)_{([p/2]}$( $d_{([p/2]}$ … $d_0$). $d_{n-1}$
else
no tag is possible.
```

Proposed permutation passable algorithm: The permutation passable gives the information about whether a unique path is available to pass a request from given source to destination or not. The request always passes from the most suitable path available; if such path is busy then the request is passed through an alternate path. If no alternate path is available then the request has to be simply dropped or said to have clash. The algorithm is:

```
Let S = source and D = destination
Switch (case)
Case identity:
S = 0 , D = 0
P = 0
For I = 1 to $2^n$ //n is size
begin
Route[S][D][p] = routing tag algorithm(S, D)
S = S+1
D = D+1
P = P+1
End //of loop
For I = 0 to p-1
begin
For J = I+1 to p-1
begin
If route[I] = route[J]
Then print clash
End // of loop j
End // of loop I
End case
Case incremental:
Input the value of S and D
P = 0
For I = 1 to $2^n$ //n is size
begin
Route[S][D][p] = routing tag algorithm(S,D)
S = S+1
If S>$2^{n-1}$ then S = 0
D = D+1
If D = $2^{n-1}$ then D = 0
P = P+1
End //of loop
For I = 0 to p-1
begin
For J = I+1 to p-1
begin
If route[I] = route[J]
Then print clash
End // of loop j
End // of loop I
End case
```

PERFORMANCE ANALYSIS OF SOME POPULAR MINS

Permutation Passable Analysis of FT (Four Tree) And QT (Quad Tree) Networks: As such it is declared earlier by the researchers that FT[4] and QT[10] network can provide full access capability even in the presence of multiple paths[6]. The performance analysis shows that there is one critical case at stage 3 where no request is passable.

Table 1: Incremental permutation measures of FT and QT

Stage	Switch /faults	Total path length	Total no of request passes	Average path length	(%) passable
	Without	20	4	5	25
	Mux	20	4	5	25
1	S1/A	20	4	5	25
	S1/B	20	4	5	25
2	S2/A	15	3	5	18
	S2/B	10	2	5	12
3	S3/A	10	2	5	12
	S3/B	0	0	0	0
4	S4/A	15	3	5	18
	S4/B	10	2	5	12
5	S5	20	4	5	25
	Demux	20	4	5	25

A: Non-critical; B: Critical case

Permutation Passable Analysis of MFT[6] Network:

Table 2: Incremental permutation measures of MFT

Stage	Switch /faults	Total path length	Total no of request passes	Average path length	(%) Passable
	Without	40	8	5	50
	Mux	40	8	5	50
1	S1/A	35	7	5	43
	S1/B	30	6	5	37
2	S2/A	30	6	5	37
	S2/B	20	4	5	25
3	S3/A	30	6	5	37
	S3/B	20	4	5	25
4	S4/A	30	6	5	37
	S4/B	20	4	5	25
5	S5	35	7	5	43
	Demux	40	8	5	50

A: Non-critical; B: Critical Case

Permutation Passable Analysis of IFT Network:

Table 3: Incremental permutation measures of IFT

Stage	Switch /faults	Total path length	Total no of request passes	Average path length	(%) passable
	Without	24	8	3	50
	Mux	24	8	3	50
1	S0/A	21	7	3	43
	S0/B	18	6	3	37
2	S0/A	21	7	3	43
	S0/B	18	6	3	37
3	S0	21	7	3	43
	Demux	24	8	3	50

A: Non-critical; B: Critical Case

It has been analyzed from the above tables that permutation passable of IFT is better than existing FT and QT, For example the permutations passable of FT and QT at SE 3 are 0 if it gets faulty in loop but in IFT minimum requests served are 37% in critical case. It has also been analyzed that IFT is better in terms of average path length in comparison to MFT, FT and QT.

COST-EFFECTIVENESS ANALYSIS

The cost of FT network is evaluated as:

- Total no of 3×3 switches = 18
- Total no of 2×2 switches = 8
- Total no of mux and demux (2:1, 1:2) = 32
- Hence cost of the network is = 258

Table 4: Comparison of cost-effectiveness

MIN	Cost
FT	$9.75\ 2^{n+1}$-54
QT	$9.75\ 2^{n+1}$-54
MFT	$9.75\ 2^{n+1}$-36
IFT	$9.75\ 2^{n+1}$-76

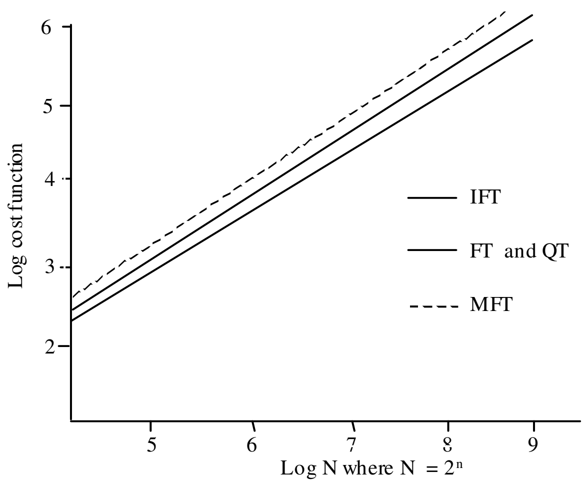

Fig. 2: Log Cost function versus log N

CONCLUSION

In this study a new Irregular Fault Tolerant Multistage Interconnection Network called IFT has been proposed and analyzed. New algorithms are designed to evaluate the performance of Irregular Fault Tolerant Multistage Interconnection Network .It has been observed from the analysis that the permutation passable of IFT is better than existing irregular FT, QT and MFT as shown in Table 1-3. It has been also observed form Table 4 and Fig. 2 that the network like IFT costs lesser in comparison to existing other popular Fault Tolerant Irregular MINs.

REFERENCES

1. Bhuyan Laxmi N., Qing Yang and P. Agrawal Dharma, 1989. Performance of multiprocessor interconnection networks. IEEE Trans. Comput., 22: 25-37. Doi: 10.1109/2.19830.
2. Kruskal, C.P. and M. Snir, 1983. Performance of multistage interconnection networks for multiprocessors. IEEE Trans. Comput., C-32: 1091-1098. Doi: 10.1109/TC.1983.1676169.
3. Patel, J.H., 1979. Processor-Memory Interconnections for multiprocessors. Computers, IEEE Transactions on, Volume: C-30, Issue: 10 771-780 , Doi: 10.1109/TC.1981.1675695
4. Bansal, P.K., K. Singh and R.C. Joshi, 1992. Routing and path length algorithm for a cost effective four tree multistage interconnection network. Int. J. Elect., 73: 107-115. Doi: 10.1080/00207219208925650.
5. Bansal, P.K., R.C. Joshi and Kuldip Singh, 1994. On a fault tolerant multistage interconnection network. Comput. Elect. Eng., 20: 205-208. Doi: 10.1016/0045-7906(94)90047-7.
6. Sandeep Sharma and P.K. Bansal, 2002. A new fault tolerant multistage interconnection network, TENCON '02. Proceedings. 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering, vol. 1,347-350, Doi: 10.1142/S0219265904001064.
7. George, B.A., P.A. Dharma and H.J. Seigel, 1987. A survey and comparison of fault-tolerant multistage interconnection networks ,*Computer*, vol. 20, no. 6, pp. 14-27, Jun., 1987: Doi: 10.1109/MC.1987.1663586.
8. Park, J.H., 2006. Two-dimensional ring-Banyan network: A high-performance fault-tolerant switching network. Elect. Lett., IEEE 2006, vol. 42,249-251: Doi: 10.1049/el:20062728.
9. Bataineh, S.M. and B.Y. Allosl, 2001. Fault-tolerant multistage interconnection network. J. Telecommun. Syst., 17: 455-472. Doi: 10.1023/A:1016779316848.
10. Bansal, P.K., K. Singh and R.C. Joshi, 1992. Quad tree: A cost-effective fault-tolerant multistage interconnection network, Proceedings of the eleventh annual joint conference of the IEEE computer and communications societies on One world through communications, Vol. 2, 860-866,http://portal.acm.org.