ISTANBUL UNIVERSITY ENGINEERING FACULTY
JOURNAL OF ELECTRICAL & ELECTRONICS

YEAR : 2002 (477-481)
VOLUME : 2
NUMBER : 2

A MULTICAST ROUTING ALGORITHM BASED ON PARALLEL BRANCHING METHOD FOR FAULTY HYPERCUBES

Salih GUNES[1], Nihat YILMAZ[1] and Novruz ALLAHVERDI [2]

[1]Selcuk University, Eng.-Arch. Fac., Electrical & Electronics Dept.,
[2]Selcuk University, Tech. Educ. Fac. Electronics & Computer Educ. Dept.,
42031-KONYA/TURKEY

E-mail: sgunes@selcuk.edu.tr , nhtylmaz@hotmail.com, noval@selcuk.edu.tr

ABSTRACT

In this study, a multicast routing algorithm based on parallel branching method has been developed for a faulty hypercube parallel processing system. The routing from the source to the destination nodes is guaranteed in shortest time with this algorithm. Going through to the destinations from the source is a parallel process at each step. The superiority of the developed algorithm over the previous studies is that the routing from the source to the destination is achieved in minimal step without restriction to the number of faulty nodes. This means that the algorithm is running independent from the number of the faulty nodes. The algorithm is simulated with a hypercube routing simulator.

Keywords: *FaultyHypercube,multicast routing,parallel branching.*

1. INTRODUCTION AND NOTATIONS

There are 2^n nodes in an n-dimensional hypercube (n-cube). The nodes are labeled from 0 to $2^n - 1$ where the labels of the neighboring nodes differ from each other with only one bit. A hypercube may be subdivided into subcubes[13]. If a hypercube has faulty node(s) or link(s) then it is called faulty hypercube. There are several studies on hypercubes. Subcube replacement and various routing algorithms are some of the examples of these studies [2,10,11,12]. The routing algorithms are inspected in two parts depending on the length of the data path, namely minimal or non-minimal. In a further step, depending on the number of the destination nodes these are also classified in three distinct titles as node-to-node, node-to-multiple nodes and node-to-all nodes routing algorithms. These three different routing algorithms have an important role especially for minimal routing algorithms. On this subject several different researchers suggested various routing algorithms [4,6,7,14,15]. The aim of the node-to-node routing algorithm is to transmit the data from the source to the destination node in minimal steps. In the routing mechanism, different types of

Received Date : 11.4.2002
Accepted Date: 31.5.2002

switching modules are used. Packet switching and circuit switching are two of these modules. Both have advantages and disadvantages depending on the way they are used. The circuit switching system is considered in this study. In this switching system, the physical path from the source to the destination is established before beginning the routing. That is, the nodes by means of which the data will be transmitted are previously identified and a communication path is established between the nodes which will be used for routing the data from the source to the destination. There is no need for the nodes to have a temporary memory. This is the superiority of the circuit switching over packet switching. Once the communication path is established the data is transmitted in real time. The total delay in this switching technique is ((Lc/B)D+L/B) where Lc is the width of the control packet sent through the established circuit, L is the length of a data packet, B is the channel-band width and D is the distance from the source to the destination node. In the total communication delay D may be omitted if Lc<<L. If the data are ordered and long enough, circuit switching is generally advantageous. The transmission failure of some other data is a disadvantage while transmitting the current data. After the data is transmitted from the source to the destination, the circuit is prepared again for other data and the entire process repeats [8].

The method followed in the algorithm is described in the next section. The developed routing algorithm and the simulator model are given in the third section. In the last section, the results of the study are discussed with respect to its advantages and disadvantages comparing with previously suggested methods on this subject.

2. THE METHOD FOLLOWED IN THE ALGORITHM

The algorithm uses some sort of arrays storing the various properties of the cubes and the nodes related to the information on the dimension of the hypercube and the faulty nodes.

The arrays used in parallel branching method are:
ways[n,1] := current node
ways[n,2] := activity of the path
node[n,1]:= the name of the node
node[n,2]:=the name of the path on the node
node[n,3]:=the previous node

A path concept is used in this algorithm. Beginning from the source node, progressing is made step by step for each loop. In the first loop, the data is transmitted from the source node to the neighboring non-faulty nodes. The related circuit is established by using paths as much as the number of the non-faulty neighboring nodes. The numbers of these paths, the source and the destination node for these paths and the activity of them are held in the array called "*ways*".

The nodes on these paths are labeled as *"this node has been reached*" , so revisiting these nodes in another loop is avoided. In the next loop, new paths are found which end in the non-faulty and unreached neighboring nodes of the current destination nodes. One of these paths contributes an additional link to the previous path. While finding these paths the visited nodes are also labeled as *"this node has been reached*" and the properties of each path are updated, but these newly found paths are not followed until all of the previous paths are inspected. After all of the previous paths are processed we obtain a group of new paths. The entire process is applied to these new paths and so on. If there are no ways out from the current paths it means that the neighboring nodes are visited in one of the previous steps. Assuring that a shortest path has been previously found which ends on one of these neighboring nodes, the newly found path is discarded. In other words, the activity of the path is assigned "*false*". If each of these paths reaches to the destination node, it is assumed that the shortest path from the source node to the destination is found and the path is stored. To store the paths from the source to the destination nodes is achieved by means of the data structure holding the previous node for each node. Using this information for each node the source is found. The path from the source to the destination is also recorded by using this information but in reverse order, beginning from the source using the neighboring node information in each node.

3. THE DEVELOPED ROUTING ALGORITHM

For a 5-dimensional hypercube, let the source node is 00000, destination nodes set is {11111,10100,11110}and the faulty nodes set is {01100, 11100, 10101, 11010, 11101, 11001, 11000, 10000, 10110, 01110, 01011}. In the first step the non-faulty neighboring nodes set from

the source node is assessed. Then the other non-faulty neighboring nodes are determined from these nodes. This operation continues until the destination node is reached. The previously reached nodes are not revisited. Once the destination node is found the communication path from the source node is determined by using the neighboring node information in each node as described in the previous section.

For the hypercube example given above the path from the source to the destination node(11111) is

00000→01000→01001→01101→01111→11111

The data route found for the node 11110 is 00000→01000→01001→01101→01111→11111 →11110

The data route found for the node 10100 is

00000→00100→10100

This routing paths are one of the shortest path for the given hypercube. The simulator image of this example is seen in Figure 1.

The multicast routing algorithm developed based on parallel branching method is given in the following.

// Multicast Routing Algorithm Based On Parallel Branching Method //

```
procedure TAnaProg.paralelbtn1Click(Sender:
TObject);
begin
ways[1,1]:=1;
ways[1,2]:=1;
ways[1,3]:=1;
node[1,1]:=source;
node[1,2]:='1';
node[1,3]:='1';
kl:
for ii:=1 to waycount do
if ways[ii] is active then
begin
degstr1:=neighbor node of  node[ways[ii,1],1];
for jj:=1 to errornodecount do if
grup1data[jj]=degstr1 then error:=1;
for jj:=1 to nodecount do if node[jj,1]=degstr1 then
error:=1;
if degstr1∉errornode cubes and  degstr1 is new
node then begin
ways[ii,2]:=0;
d:=1;
change:=change+1;
nodecount:=nodecount+1;
node[nodecount,1]:=degstr1;
node[nodecount,2]:=inttostr(waycount+change);
node[nodecount,3]:=inttostr(ways[ii,1]);
ways[waycount+change,1]:=nodecount;
ways[waycount+change,2]:=1;
ways[waycount+change,3]:=jj;
if degstr1 is a target then begin writeshortway;
found:=found+1;end
if found=count of target then begin
edit1.text:='found'; goto eet1;end;
end;
end;  //ÝF
end;  //FOR
if d=0 then ways[ii,2]:=0;
end; //FOR
waycount:=waycount+change;
if d≠0 then begin change:=0;d:=0;goto kl;end;
eet1:
end;
```

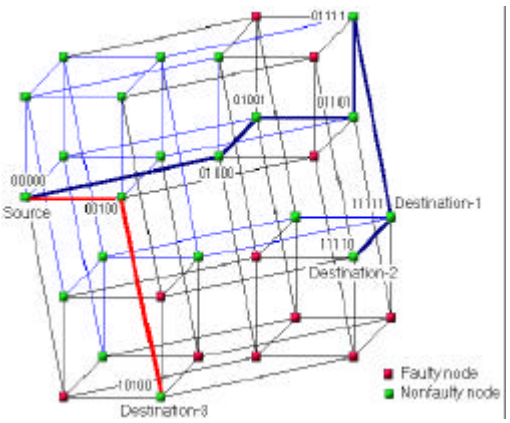


Figure 1. The Simulator Image of the Example Implemented for A 5-Dimensional Hypercube

4. RESULTS AND DISCUSSION

With the developed algorithm the minimal route from the source node to the destination nodes are guaranteed. While doing this no restriction is assumed for the number of the faulty nodes which was a lack of the previous studies. In Figure 2, for 6 previously defined destination node, the needed number of operations for different number of faulty nodes in one-to-multiple nodes routing algorithm is extracted and the performance analysis of the algorithm is described [16].

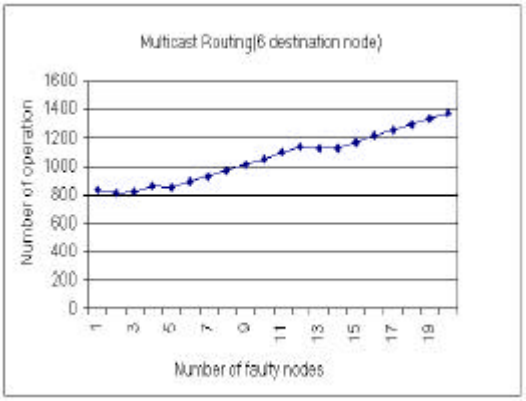


Figure 2. The change of the number of faulty nodes against the number of operations

Again, this time assuming the faulty node set is constant, the change of the number of the operations when increasing the number of the source nodes is given in Figure 3. For these routing algorithms the source node is also 000000. The set for 6-destination is assumed as = { 100000, 010000, 000100, 100001, 010010, 0100001}

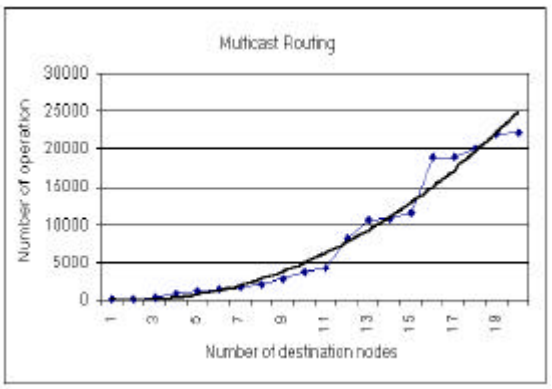


Figure 3. The change of the number of operations against the number of destination nodes

Comparing with the multicast routing algorithm based on cube algebra developed by us, the parallel branching method has better performance. The cube algebra method needs more

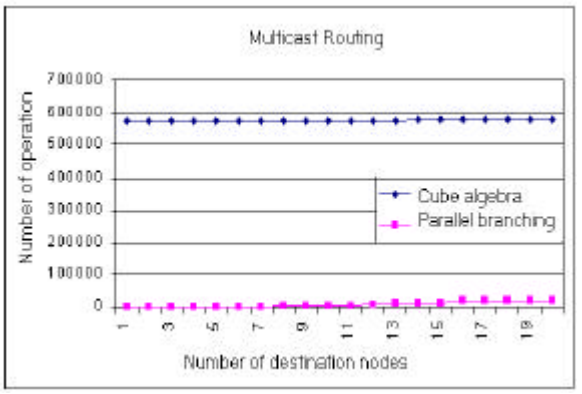


Figure 4. Comparison of Cube algebra and parallel branching methods that the change of the number of operations against the numb er of destination nodes

operations for the example having the same number of faulty nodes.(Figure 4) Moreover, parallel branching algorithm is faster than cube algebra method. Both of these methods have the same advantage of guaranteeing the minimal path independent from the number of faulty nodes. The polynomial (cube algebra) curve is the average of the cube algebra curve. Various routing algorithms are being studied by means of the written routing simulator program.

REFERENCES

[1] Abali, B, Özgüner, F. and Aykanat, C. "Dynamic Subcube Allocation in Hypercube Multiprocessor", *The Proceedings of the Fourth Conference on Hypercubes Concurrent Computers and Applications*, pp.269-272, 1989.

[2] Allahverdi, N.M., et all. "A Fault Tolerant Routing Algorithm Based on Cube Algebra for Hypercube Systems". *Journal of Systems Architecture (JSA)*. Vol.46, Iss.2 (January), pp.201-205, 2000.

[3] Burch, H. J. and Ercal, F. "A Fast Algorithm for Complete Subcube Recognition", *Proceedings of the 1997 Int. Symp. On Parallel Architectures, Algorithms and Networks*, 85-90, 1997.

[4] Chan, M.Y. and Lee, S.J. "Fault-Tolerant Embedding of Complete Binary Trees in Hypercubes". *IEEE Transactions on Parallel Distributed Systems*. Vol.4, No:3, pp.277-288, 1993.

[5] Chang, Y. and Bhuyan L. "Subcube Fault Tolerance in Hypercube Multicomputers". *IEEE Transactions on Comput*. Vol.44, No:9, pp.1108-1120, 1995.

[6] Chen, M.S. and Shin, K.G. "Dept-First Search Approach for Fault-Tolerant Routing in Hypercube Multicomputers". *IEEE Transactions on Parallel Distributed Systems*. Vol.1, No:2, pp.152-159, 1990.

[7]Chiu, G.M. and Wu, S.P. "A Fault-Tolerant Routing Strategy in Hypercube Multicomputers". *IEEE Transactions on Comput*.. Vol.45, No:2, pp.143-155, 1996.

[8] Duato, J., Yalamachili, S. and Ni, L. "Interconnection Networks". *IEEE Computer Society Press, Piscataway*, 1997.

[9] Güneº, S. et all. "Simulation of New Fast Algorithm Basaed on Cube Algebra for Subcube Allocation in Faulty Hypercubes". *Proc. of The Second Int. Symp. on Math. & Com. Appl*, Baku, pp.169-175, 1999.

[10] Güneº, S. et all. "Realization of Nonminimal Routing Algorithm in Faulty Hypercube Parallel Processing System With Using Cube Algebra". *ELECO'99*, pp.290-293, 1999.

[11] Lan, Y. "An Adaptive Fault-Tolerant Routing Algorithm for Hypercube Multicomputers". *IEEE Transactions on Parallel and Distributed Systems*. Vol.6, No:11, pp.147-1152, 1995.

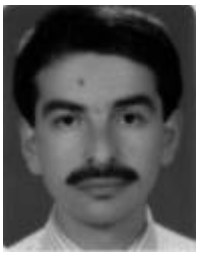
[12] Lee, T.C. and Hayes, J.P. "A Fault-Tolerant Communication Scheme for Hypercube Computers". *IEEE Transactions on Computers*. Vol.41, No:10, pp.1242-1256, 1992.

[13] Saad, Y. and Schultz, M. H. "Topological Properties of Hypercubes". *IEEE Transactions on Computers*.. Vol.C-37, No:7, pp.867-872, 1988.

[14] Raghavendra, C.S., Yang, P.J. and Tien, S.B. "Free-Dimensions An Approach to Achieving Fault Tolerance in Hypercube". *IEEE Transactions on Computers*. Vol.44, No:9, pp.1152-1156.

[15] Tien, S.B. and Raghavendra, C.S. "Algorithms and Bounds for Shortest Paths Diameter in Faulty Hypercubes". *IEEE Transactions on Parallel and Distributed Systems*. Vol.4, No:6, pp.713-718, 1993.

[16] S. Güneþ N. Yýlmaz and A. Öztürk, "A Fault –Tolerant Multicast Routing Algorithm Based on Cube Algebra for Hypercube Multicomputers", *MELECON 2000*, pp: 758-761, Cyprus.



Salih Günes was born Kadýnhaný-Konya on July, 1967. He received the B.Sc. and M.Sc. degrees in Electronics Engineering Department from Graduate School of Natural and Applied Sciences at Erciyes University in 1989, 1993 respectively. He received Ph.D. degree in Electrical-Electronics Engineering Department from Graduate School of Natural and Applied Sciences at Selçuk University in April 2000. During 1990-2000 he worked as a research assistant in the Selcuk University. He is now assistant Professor of Electrical-Electronics Engineering Department of Selçuk University. His research interest are in the area of computer hardware, signal and image processing, parallel and distributed computing, neural network,fault-tolerant computing.

Nihat Yilmaz received his BS in Electrical-Electronics Engineering from Selçuk University in 1996, his MS in Electrical-Electronics Engineering Science from the Graduate School of Natural and Applied Sciences at Selçuk University in 1998. He is currently a PhD student in Graduate School of Natural and Applied Sciences at Selçuk University. Currently he is a research assistant with the Selçuk University. His research interest are in the area of computer hardware, signal and image processing, parallel and distributed computing, neural network, fault-tolerant computing.

Novruz Allahverdi was born in Azerbaijan in 1948. He graduated from The Department of Computers and Systems of Azerbaijan Technical University, Baku in 1972. He received his Ph.D degree in Computer Science from the Moscow Power Institute, Moscow, Russia in 1979. He has co-authored 10 books and authored (co-authored) over 70 papers in the areas of parallel systems for the digital signal processing, especially for FFT type algorithm, hypercube systems, artificial intelligence, especially expert systems. Currently he works at Selçuk University, Konya, Turkey where he is the head of Electronics and Computer Education Department, Faculty of Technical Education.