Journal of Computer Science 2 (2): 124-126, 2006
ISSN 1549-3636

Adaptive Fault Tolerant Routing Algorithm for Tree-Hypercube Multicomputer

Qatawneh Mohammad
Department of Computer Science, Al-Zaytoonah University of Jordan
P.O.Box 130 Amman (11733) Jordan

Abstract: A Connected tree-hypercube with faulty links and/or nodes is called injured tree-hypercube. To enable any non faulty node to communicate with any other non faulty node in an injured tree-hypercube, the information on component failures has to be made available to non faulty nodes to route message around the faulty components. We proposed an adaptive fault tolerant routing algorithm for an injured tree-hypercube in which requires each node to know only the condition of it's own links. This routing algorithm is shown to be capable of routing messages successfully in an injured tree-hypercube as long as the number of faulty components links and/or nodes is equal d (depth).

Key words: Fault-tolerant routing, Injured and regular tree-hypercube, hypercube

INTRODUCTION

The increased emphasis on fault-tolerance and reliability has made distributed system architecture particularly attractive. An important component of a distributed system is the system topology. The system topology defines the interprocessor communication architecture. Also, there are well-defined relationships between the system topology and the message delay, the routing algorithms and fault-tolerance. Specially, the message delay may be directly proportional to the internode distance.

Efficient routing of message is a key to the performance of a multicomputer system[1-3]. Especially, the increasing use of multicomputer systems for reliability-critical applications has made it essential to design fault-tolerant routing strategies for such systems. By fault-tolerant routing, we mean the successful routing of messages between any pair of non faulty nodes in the presence of faulty components (link and/or nodes).

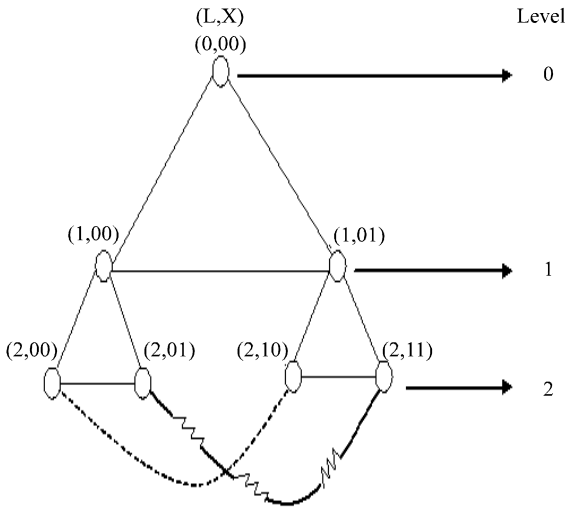
A connected tree-hypercube TH(s,d) with faulty components is called an injured tree-hypercube, whereas a tree hypercube without faulty components is called a regular tree-hypercube. In regular tree-hypercube, the routing of a message can be viewed as a sequence of changes made on the source address label. These changes are done at every intermediate node in the path. When the message is received by an intermediate node, it will consider itself as a new source. However, this scheme of routing becomes invalid in an injured tree-hypercube, since the message may be routed to a faulty component. In order to enable non faulty nodes in an injured tree-hypercube to communicate with one another, enough network information has to be incorporated into either the message to be routed or each node in the network so as to route messages around the faulty components. Hence it is important to develop routing scheme which require each node to keep only the failure information essential for making correct routing decisions. For this reasons above, we shall develop a routing scheme which requires each node to know only the condition of it's own links. This scheme is proven to be capable of routing a messages between any pair of non faulty nodes as long as the total number of faulty components is equal d in TH(s,d). Figure 1 shows two tree-hypercubes TH(2,2) and TH(2,3).

A TH(s,d) network[4] is constructed by taking a full tree of degree s^2 and of depth d. Levels of the tree are numbered 0,.., d. Each level i has s^i nodes representing processing elements as a hypercube. Thus, nodes at level i constitute (i log s)-cube. At each level nodes are labeled in binary from 0 to s^k-1, where k=level i. Each node in TH is identified by a pair (L, X) of its level number L and its cube address X. The total number of nodes in TH(s,d) is $N= (s^{d+1}-1)/(s-1)$.

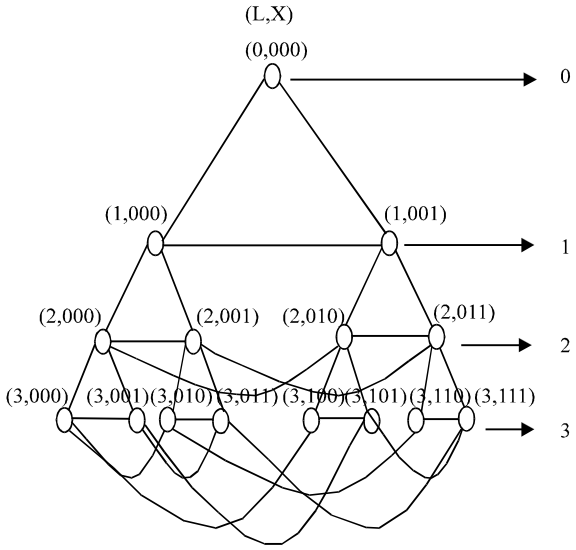
Definition[4]: Tree-hypercubes can be formally defined as follows:

The levels and nodes are labeled as above. For every level $0<L<d$, each node (L,X) in level L, where $X= X_{(d\ logs)-1}\ldots\ldots X_0$ is adjacent to the following s children nodes at level L+1 (L+1, X.a) for a = 0,…..,s-1, if L<d (X.a is the concatenation of the label X and the binary digit(s) a) and to the parent node (L-1, $X_{(d\ log\ s)-1}\ldots\ldots X_{log\ s)}$. Node (L,X) is also adjacent to L log s nodes (L,Y) in the same level where Y's address fifers from X in exactly one bit position.

Corresponding Author: Dr. Qatawneh Mohammad, Department of Computer Science, Al-Zaytoonah University of Jordan, P.O. Box 130 Amman (11733) Jordan, Tel: 0096279550220



(a)



(b)

Fig. 1: Tree-hypercube (a) TH(2,2) and (b) TH(2,3)

Routing with information on local link failures: Now, we develop and analyze a fault tolerant algorithm. This algorithm is proven to be capable of routing a message between any pair of non faulty nodes as long as the total number of faulty components is equal d. The algorithm requires every node to know only the condition of its own links.

Adaptive fault-tolerant routing algorithm: Before describing algorithm, it's necessary to introduce the following definition.

Definition 1: Tree-Hypercube TH(s,d) contains $N=(s^{d+1}-1)/(s-1)$ nodes and $N-1+\sum^{d}_{i=0}(S^{d} * \log_2 S^{d})/2$ links, since the degree of each node in TH(s,d) is $E=E_t+E_c$, where E_t is a set of the tree edge and E_c is the hypercube edges.

$E_t = [t_0,\ldots,t_{m-1}]$
Where m = 2 if L=0, m = 3 if 0 < L< d, or m= 1 if L=d.
$E_c = [C_0,\ldots..,C_n]$, where n = Level number for every level > 0.
Fault – Tolerant Routing for TH(s,d)
{Source node X with address (L_s, $X_{(d \log s)-1}\ldots..X_0$).
{Destination node Y with address(L_d, $Y_{(d \log s)-1}\ldots..Y_0$).
Begin
i = $|L_d-L_s|$*log s-1; (used when $L_s < L_d$ to find the position of digit(s) of Y to concatenate to X)

Case 1: (Ls = L_d) {both on the same level}.
Tag := src ⊕ dest; starting with the most significant bit of tag, let i number of first
1 in tag.
IF (ptr[i] is not faulty) then
Send to (L_s,ptr[i]^.node_number);
ELSE
Remove ptr[i] from the coordinate sequence and using tag check for the first non fault ptr[x] in the coordinate and send to (L_s,ptr[x]^.node_number)
ELSE
Use tree link (up) only for one hope if its not faulty else down for one hop.

Case 2: ($L_s < L_d$) downward
Send to { L_{s+1}, $Y_{(d \log s)-1}\ldots..X_0Y_i$)
ELSE
Check for the first non faulty ptr[x] (tree link) in the coordinate sequence and send to (L_{s+1}, ptr[x]^node_number).
ELSE
Use n-cube links only for one hope.

Case 3: ($L_s > L_d$) Upward
IF (tree link (up) is not faulty)
THEN send to (L_{s-1}, $X_{(d \log s)-1}\ldots..X_1$).
ELSE
Use cube link only for one hope.
End.

The worst case of algorithm needs d+2 steps to send the message from X to Y in an injured tree-hypercubes. It can be shown that the diameter in injured TH(s,d) greater than the diameter of regular TH(s,d) by 2. This is refer to the addition of the n-cube links to the binary tree which creates multiple disjoint path between every pair of nodes.

CONCLUSION

In this study, we have proposed and analyzed an adaptive fault tolerant routing algorithm to route a message in an injured tree-hypercubes multiconputer. This algorithm is very powerful and route a message via an optimal path from the source to destination. This is refer to the addition of the n-cube links to the binary tree which creates multiple disjoint path between every pair of nodes. It requires each node to know only the condition of its own links. Due to its simplicity and/or power, the fault – tolerant routing algorithm derived in this study have potential used on tree-hypercubes multicomputers.

REFERENCES

1. Chen, M.S., 1990. Adaptive fault-tolerant routing in hypercube multicomputer. IEEE Trans. Comput., c-39: 1406-1416.
2. Neison, V.P., 1990. Fault-tolerant computing fundamental concepts. IEEE Trans. Comput., 33: 19-25.
3. Pradhan, D.K. and S.M. Reddy, 1982. A fault-tolerant communication architecture for distributed systems. IEEE Trans. Comput., c-31: 863-870.
4. Al-Omari, M. and H. Abu-Salem, 1997. Tree-hypercubes: A multiprocessor interconnection topology. Abhath Al-Yarmouk, 6: 9-24.