EasyChair Preprint

№ 3503

Disjoint Multipath Routing and Failure Recovery by Maintaining the Colored Trees

R Sudha Abirami

May 30, 2020

DISJOINT MULTIPATH ROUTING AND FAILURE RECOVERY BY MAINTAINING THE COLORED TREES

Mrs.R.Sudha Abirami
Assistant Professor, Thassim Beevi Abdul Kader College for Women, Kilakarai
sudha.tbakc@gmail.com, 63799 33676

Abstract

In this paper the Coloured trees are constructed for routing the packets along link or node disjoint paths. But maintaining the trees after link/node failure is difficult. To overcome this problem an algorithm named, SCT is developed in this paper that construct and maintains the coloured trees in an efficient manner and reduces the message overhead when compared with existing techniques. For each and every node in the network two disjoint paths are created. When a node failure occurs in one path it then reroutes the packets along another path. It requires fewer messages for reconstructing the tree after node failures. The above mentioned improvements are obtained by exploiting the relationship between DFS numbering, low-point values, and the potentials employed for maintaining partial ordering of nodes.

Keywords: low-point value, multipath routing, reconfiguration

1. INTRODUCTION

The term multipath routing [1] has been used to describe the class of routing mechanisms that allow the establishment of multiple paths between source and destination. Classical multipath routing has been explored for two reasons. The first is load balancing [2]: traffic between a source-destination pair is split across multiple (partially or completely) disjoint paths. The second use of multipath routing is to increase the likelihood of reliable data delivery. In these approaches, multiple copies of data are sent along different paths, allowing for resilience to failure of a certain number of paths [3]. In general, the multiple paths from a source to a destination may have common links (or nodes) as long as the shared links (or nodes) have sufficient resources.

To improve the transmission reliability and avoid shared-link (or node) failures, the multiple paths can be selected to be link-or node-disjoint. In this case, the MPR approach is referred to as disjoint multipath routing (DMPR). DMPR provides better robustness compared to the generic MPR. If multiple paths are employed for increased throughput and for secure communication [5], then the data may be split over multiple paths.

To reduce the routing table overhead, hence reduce lookup time, a novel multipath routing strategy called coloured trees (CT) was developed [6]. Every node in the network has two preferred neighbours to the destination: red and blue. A packet transmitted from a source is marked with one of the two colours. An intermediate node that receives the packet forwards it to its preferred neighbour based on the colour of the packet. Thus, the routing table at a node has only two entries (for every destination node). The two paths from a given source to the drain on the two trees are link/node-disjoint.

A major limitation of the previous path augmentation technique [7] is that the maintenance of the trees is difficult in the presence of failures, as paths are augmented in a sequential fashion. The failure of a node/link may result in invalidating the paths for several nodes. While the trees may be reconstructed entirely after the failure, such a total reconstruction of the trees is both unnecessary and results in a high overhead. As all algorithms based on path augmentation suffer from the above limitation, this paper develops a new algorithm, referred to as the SCT algorithm. The SCT algorithm offers four advantages compared to the earlier algorithms based on path augmentation: (1) the SCT algorithm reduces the message overhead required to construct the trees by 40%; (2) the average path lengths obtained using the SCT algorithm is lesser than those obtained using earlier approaches; (3) the number of nodes whose paths are affected by a failure under the SCT algorithm will never be more than those affected by path augmentation based approaches.

2. ROUTING TREE CONSTRUCTION

The SCT algorithm works in three phases for constructing the routing trees with maximum disjointedness: (1) distributed DFS numbering and low-point computation; (2) distributed layering; and (3) selection of left (blue) and right (red) forwarding nodes. This paper first produces the path to satisfy the CT-ND constraint.

2.1 DETERMINING NODE DISJOINT PATH

2.1.1 Numbering and Low point Computation Phase:

The first phase of the algorithm is to assign DFS indices to all the nodes and compute the low-point value and path for each node. We employ the generalized low-point concept, developed in [7], in the SCT.

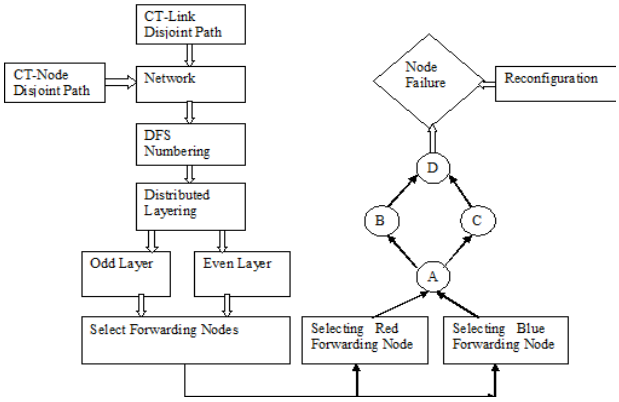

Fig.1 Phases for Disjoint Path Selection

Traditionally, the low-point value of a node x is defined as the smallest DFS index node that may be reached from x by traversing DFS tree edges and one back edge.

DFS(parent; n; currdfs)

1. if dfs[n] > 0 return currdfs;

2. dfs[n] = currdfs; dfsparent[n] = parent;

currdfs = currdfs + 1;

3. for every neighbor i ≠ parent of n do:

3.A. currdfs = DFS(n, i, currdfs);

3.B. if (dfs[i] < dfs[n]) and (dfs[i] ≤glpv[n])

3.B.i. glpv[n] = dfs[i]; glpn[n] = i; glpd[n] = Cni;

3.C. else if (dfs[i] > dfs[n]) and (glpv[i] < glpv[n])

3.C.i. glpv[n] = glpv[i]; glpn[n] = i; glpd[n] = glpd[i] + Cni;

3.D. else if (dfs[i] > dfs[n]) and (glpv[i] = glpv[n]) and (glpd[i] < glpd[n] - Cni)

3.D.i. glpn[n] = i; glpd[n] = glpd[i] + Cni;

4. return currdfs;

Fig.2 Algorithm to assign DFS-indices to the nodes and compute glpv value and neighbour

The generalized low-point value (GLPV) of a node x is the smallest DFS index node that may be reached from x by traversing nodes in the increasing order of their DFS indices except the last hop. The generalized low-point neighbour (GLPN) of a node x is the neighbour of x in its generalized low-point path. The steps of the DFS numbering phase are shown in the given algorithm. The algorithm to assign the DFS-indices and compute the GLPV and GLPN is shown in Fig 3. The DFS-indices of all the nodes are first initialized to -1.

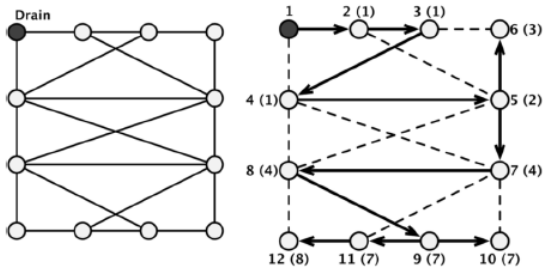

Fig.3 An example network and DFS numbering

Consider the network shown in Figure 3. We will use this network to show the working of the entire algorithm. The network in Figure 3 is two-node-connected. The number next to a node represents its DFS index and the number in parenthesis denotes the GLPV. The solid lines represent the DFS tree edges. In a two-node-connected network, the GLPV of a node is strictly less than the DFS index of its parent.

2.1.2 Layering Phase:

The second phase of the SCT algorithm is to arrange the nodes into ODD and EVEN layers. To compute the layer for a node, we define a term called potential – a bound on the GLPV of a node to be present in the same layer.The potential of a node x, denoted by pot(x), is computed as follows for satisfying the CT-ND constraint:

$$pot(x)=\begin{cases} pot(p(x)) & \text{if } glpv(x) < pot(p(x)); \\ dfs(p(x)) & \text{otherwise} \end{cases}$$

where p(x) is the identifier of the DFS parent of x.

The layer of a node is assigned similar to the potential value as below.

$$l(x) = \begin{cases} l(p(x)) & \text{if } glpv(x) < pot(p(x)); \\ \sim l(p(x)) & \text{otherwise} \end{cases}$$

where ~l(p(x)) denotes the negation of the layer of node p(x).

If l(p(x)) = ODD, then ~ l(p(x)) = EVEN and vice versa.

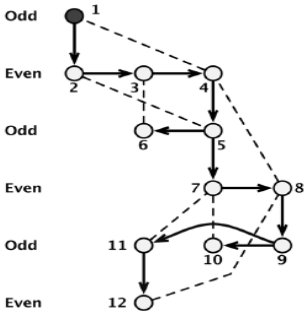

Fig.4 Placement of nodes at different layers

An exception to the above rule is for nodes whose low point value equals 1. All the nodes whose low point value equals 1 are placed in the layer next to drain (i.e. EVEN layer).

2.1.3 Red and Blue Forwarding Node Selection:

The third phase of the SCT algorithm is to select the left (red) and right (blue) forwarding nodes, referred to as lfn and rfn, respectively. Irrespective of how the layering approach is performed, the selection method for the left and right forwarding nodes is the same. In the distributed layering phase we always place the drain at the first layer (ODD).

If node x is in the EVEN layer, then

lfn(x) = p(x) and rfn(x) = glpn(x).

If node x is in the ODD layer, then

lfn(x) = glpn(x) and rfn(x) = p(x)

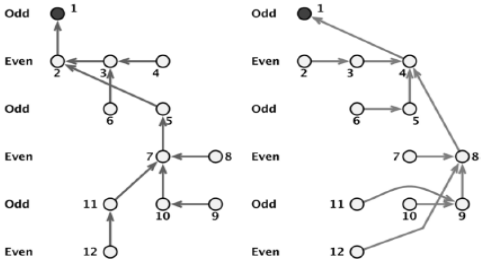

Fig.5 Left (Red) and Right (Blue) trees obtained from the layered structure

We arrange the nodes in the even layer in the increasing order of their DFS numbers from left to right. Therefore, at the even layer we have the low-point path in the right direction and in the left direction at the ODD layer.

2.2 DETERMINING LINK DISJOINT PATH

The first and the third steps of the algorithm are the same as that of the CT-ND version. The conditions for assigning layers are modified as follows. For satisfying the CT-LD constraint, a node x takes the potential of its parent if its GLPV is less than or equal to the potential of its parent, otherwise its potential is the DFS index of its parent.

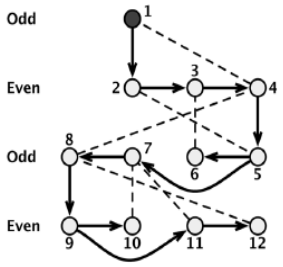

Fig.6 Placement of nodes at different layers to satisfy the CT-LD constraint.

The potential of a node x is computed as follows for satisfying the CT-LD constraint:

$$pot(x) = \begin{cases} pot(p(x)) & \text{if } glpv(x) \le pot(p(x)); \\ dfs(p(x)) & \text{otherwise} \end{cases}$$

The layer of a node is assigned as

$$l(x) = \begin{cases} l(p(x)) & \text{if } glpv(x) \le pot(p(x)); \\ \sim l(p(x)) & \text{otherwise} \end{cases}$$

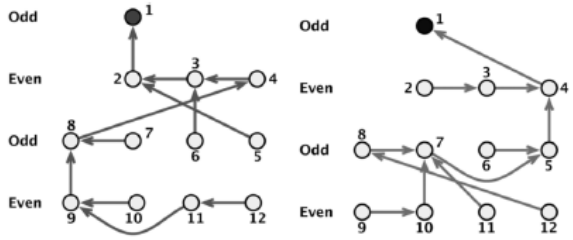

Fig.7 Left (Red) and Right (Blue) trees obtained from the layered structure

For the layered tree constructed in Figure 6 for the CT-LD case, the resulting red and blue trees are shown in Figure 7. Observe that the path from any node to the root on the two trees are only link-

disjoint, even though the network is two node-connected.

3. RECONFIGURATION PROCESS

When a node fails in the network, all the nodes that derived its DFS numbers will have their DFS index invalidated, hence the left and right forwarding nodes are also invalidated.

The children of the failed node initiate the invalidation message to their children successively. Therefore, the number of nodes that lose their forwarding nodes due to the SCT algorithm will never be higher than those that lost their forwarding nodes in the distributed or centralized path augmentation approaches implemented

To reconstruct the tree in the SCT algorithm, the DFS parent of the failed node will restart the DFS numbering phase. Note that the path augmentation technique would also start in a similar fashion, however would require the second phase of path augmentation which is an expensive (sequential) process compared to the (parallel) layering process.

The parent of the failed node reinitiates the DFS numbering procedure. However, it must be noted that the DFS numbering step must always finish with the drain as there may be nodes that are not reachable from the parent of the failed node, but may be reachable from its ancestors. Note that in the above procedure, it is relatively easier to implement the DFS renumbering while identifying the node to restart the path augmentation is harder, particularly in a distributed fashion. The SCT algorithm simplifies the reconfiguration procedure by exploiting the relationship between the DFS indices, low point values, and the potential of a node for partial ordering. Therefore, all paths augmented in the network will be discarded. Now, the trees have to be reconstructed completely even though the DFS number and GLPV of many nodes remain unchanged.

4. PERFORMANCE EVALUATION

We compare the above performance metrics obtained using the SCT algorithm with the results obtained from the existing distributed linear-time algorithm [7] based on path augmentation, indexed as "PATH" in the results.

The SCT algorithm performs better in the context of maintaining the CTs under node failures Fig. 9 compares the average number of nodes that gets affected under single node failure in computing the CTs using the path augmentation technique and the SCT algorithm. The SCT algorithm provides less convergence time and less message overhead when compared to previous path augmentation technique.

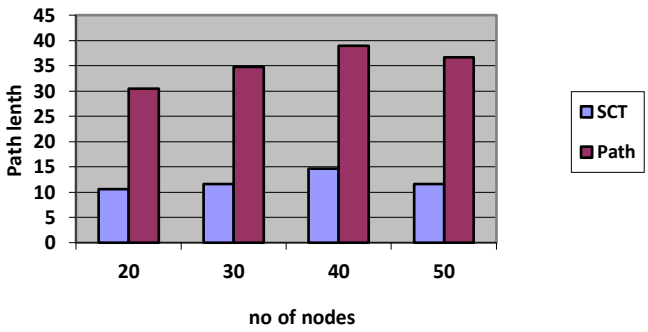

Fig.8 Comparison of path lengths obtained using the SCT algorithm and the existing linear time algorithm

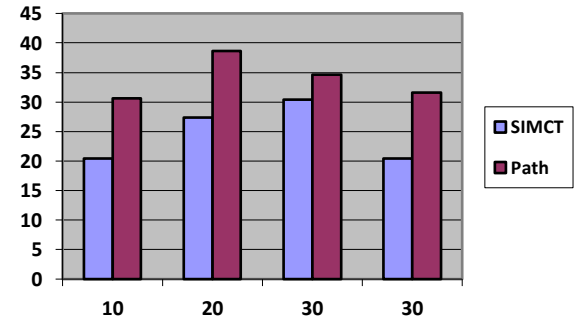

Fig.9 Comparison of average number of nodes that get affected under single node failure.

5. CONCLUSION

In this article we propose a new scheme for disjoint multipath routing in an efficient manner by creating two routing trees. In this tree every node in the network has two disjoint paths namely red forwarding path and right forwarding path. Any node failure in the network does not affect the entire tree. For that the SCT algorithm is created for constructing the colored trees with maximum disjointedness. The minimum cost path is also calculated among the two disjoint paths based on the hop count from each node to the drain. The SCT algorithm reduces the message overhead by 40% compared to the best known linear-time distributed CT approach. The performance of the SCT algorithm is found to be always better than

the existing coloured tree based algorithms for disjoint multipath routing.

REFERENCES

[1] P. P. Pham and S. Perreau, "Performance analysis of reactive shortest path and multipath routing mechanism with load balance," in Proceedings of IEEE INFOCOM, March-April 2003, vol. 1, pp. 251–259.

[2] Y. Ganjali and A. Keshavarzian, "Load balancing in ad hoc networks: Single-path routing vs. multi-path routing," in Proceedings of IEEE INFOCOM, March 2004, vol. 2, pp. 1120–1125.

[3] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly resilient energy-efficient multipath routing in wireless sensor networks," ACK SIGMOBILE Mobile Computing and Communications Review, vol. 4, no. 5, pp. 11–25, 2001.

[4] S. Murthy and J. J. Garcia-Luna-Aceves, "Congestion-oriented shortest multipath routing," in Proceedings of IEEE INFOCOM, March 1996, vol. 3, pp. 1028–1036.

[5] W. Lou, W. Liu, and Y. Fang, "A simulation study of security performance using multipath routing in ad hoc networks," in IEEE Vehicular Technology Conference, October 2003, vol. 3, pp. 2142–2146.

[6] S. Ramasubramanian, H. Krishnamoorthy, and M. Krunz, "Disjoint multipath routing using colored trees," Accepted for publication in Elsevier COMNET, vol. 51, no. 8, pp. 2163–2180, June 2007.

[7] S. Ramasubramanian, M. Harkara, and M. Krunz, "Linear time distributed construction of colored trees for disjoint multipath routing," Accepted for publication in Elsevier COMNET, vol. 51, no. 10, pp.2854–2866, July 2007.

[8] M. Medard, R.A. Barry, S.G. Finn, and R.G. Gallager, "Redundant trees for preplanned recovery in arrbitrary vertex- redundant or edge redundant graphs," IEEE/ACM Transactions on Networking, vol. 7, no.5, pp. 641–652, October 1999.

[9] G. Xue, L. Chen, and K. Thulasiraman, "Quality of service and quality of protection issues in preplanned recovery schemes using redundant trees," in IEEE Journal on Selected Areas in Communications, 2003,vol. 21, pp. 1332–1345.

[10] W. Zhang, G. Xue, J. Tang, and K. Thulasiraman, "Linear time construction of redundant trees for recovery schemes enhancing qop and qos," in Proceedings of IEEE INFOCOM, March 2005, pp. 2702–2710.

[11] G. Jayavelu, S. Ramasubramanian, and O. Younis. "Maintaining colored trees for disjoint multipath routing under node failures." IEEE/ACM Transactions on Networking, 17(1):346–359, 2009.

[12] Y. Lee, A.L.N Reddy. "Disjoint Multi-Path Routing and Failure Recovery." In Tech. Report TAMU-ECE-2009-06, Texas A& M University, June 2009.