Multicast Automatic Protection Switching in Arbitrary Redundant Graphs.

Robert G. Gallager†, Muriel Médard‡, Richard A. Barry‡, Steven G. Finn‡
† MIT LIDS, Building 35, 77 Massachusetts Avenue, Cambridge, MA 02139
gallager@lids.mit.edu
‡ MIT Lincoln Laboratory, 244 Wood Street, Lexington, MA 02173
medard@ll.mit.edu, finn@ll.mit.edu, ribarry@ll.mit.edu

Abstract

We present a new algorithm for automatic protection switching (APS) which creates node (edge) redundant trees on any node (edge)-redundant network. These trees are desirable for performing multicasting with APS. Our algorithm is based on constructing trees with appropriate associated directions. The algorithm gives great flexibility in the choice of trees.

1 Introduction.

The increasing reliance upon data networks has motivated research in the area of network reliability. In particular, for high speed networks, rapid recovery from failure is important, as even a short down time may entail the loss of much data. Self-healing networks, which restore their functionality autonomously in case of a failure, have therefore been the subject of an active area of research and are rapidly coming into widespread use. For example, self-healing features are incorporated in the SONET protocols. Another issue of increasing importance in networks is that of multicasting. Multicasting is already a common application in DoD networks and is increasingly important in civilian high-speed networks. In networks which allow duplication at nodes, a link need only carry at most a single copy of a multicast signal. Most networks, e.g. SONET, ATM, IP, and all-optical networks, allow such duplication. In such networks, multicasting is preferably performed with trees.

In this paper, we present a protection mechanism for arbitrary edge (node) redundant graphs using trees. To ensure very rapid failure recovery, we concentrate on automatic protection switching (APS), in which response to a failure is pre-planned. APS requires pre-planned spare capacity, since restoration is "hard-wired", but offers very rapid failure recovery. For example, with mechanical add-drop multiplexers (ADMs), the restoration time is of the order of tens of ms ([7]). Very rapid switching, of the order of μs, may be achieved with acousto- or electro-optic switches, e.g. [8, 4]. In this paper, we do not address the issue of excess capacity, only that of the topology of the network. For instance, in fiber optic networks, there is usually excess capacity available in the fiber but the cost of laying down the fiber is very high.

The design of SONET networks with APS has usually been considered in terms of the building blocks of self-healing rings (SHRs), possibly combined with diversity protection (DP) ([5, pp. 315–325]). Unidirectional Path Switched Rings (UPSR) use *path protection* where a back-up stream is simultaneously sent over a path disjoint from that of the primary stream. In the event of a failure, a node will switch from listening to the primary stream to listening to the back-up stream. Alternatively, Bi-directional Line Switched Rings (BLSR) use *loop-back protection* where in the event of a failure, the primary stream is redirected in the opposite direction it was originally traversing until it reaches the other side of the failure, where is rejoins its original route. Path protection is generally faster but in general requires more spare capacity than loopback.

However, interconnected rings place topological restrictions on the design of the network thereby possibly increasing cost. In addition, interconnected rings do not provide end-to-end path protection. Rather, each ring is individually protected. Although sufficient for link failure, failure of a node interconnecting two rings may require time consuming re-routing.

Menger's theorem insures the existence of edge (node) disjoint paths between any source and any destination in an edge (node) redundant graph [12]. Path protection schemes using this fact have been proposed usually associated with some sort of shortest path selection, see e.g. [9, 10]. However, when used for multicasting, the union of these disjoint paths does not guarantee trees.

Since UPSR uses disjoint primary and back-up paths, it is natural to consider using disjoint spanning trees for multicasting in arbitrary redundant graphs. Reference [11] considers a pair of disjoint undirected trees, but such an approach is not applicable to arbitrary redundant graphs. Disjoint directed trees (or branchings) ([6]) exist in any redundant graph. However, when we model bidirectional communication links as two arcs with reverse directions, the loss of a link entails the loss of two arcs which share their end nodes. Hence, arbitrary disjoint directed trees do not guarantee protection against a link (node) failure.

In ([1, 2]), we presented an algorithm to construct disjoint directed trees which do guarantee protection against edge (node) failures on arbitrary edge (node) redundant graphs. Having two such redundant trees was first presented in [3] for edge redundant graphs. In this paper, we present a generalization of ([1, 2]) which allows the construction of a larger variety of redundant trees than in ([1, 2, 3])

First, in Section 2, we present a motivating example. The algorithm is presented in Section 3. The paper ends with a summary and conclusions in Section 4.

2 APS Multicasting Protection on an Example Network

Consider the network shown in figure 1 consisting of switching nodes and bi-directional communication links. We use the term edge and link interchangeably whereas an arc refers to one direction of a link. Note that the example network is edge and node redundant in that failure of any edge or node leaves the network connected.

In this example, there is a broadcast communication from source node s to all other nodes. We present here the variety of protection mechanism discussed above.

First consider using Menger's theorem to construct edge and node disjoint paths from s to all other nodes. Clearly, unless the primary paths are chosen carefully, the resulting union of the paths will not produce a tree. Even if a tree was chosen for the primary communication, the union of the back-up paths would not necessarily form a tree. Figure 1.a shows a pair of vertex-disjoint paths between the source and vertex a and a pair of vertex-disjoint paths between the source and vertex b. The two primary paths together form a directed tree, but the two secondary paths do not form a directed tree.

Next consider using edge disjoint spanning trees. Figure 1.b shows two edge-disjoint spanning trees. The topological constraint in [11] is, however, 4-connectedness, which is greater than the edge redundancy requirement, i.e. 2-connectedness. A simple example of an edge-redundant graph which does not allow edge-disjoint spanning trees is given in figure 2. If $[c, d]$ is in the primary tree, then c must be reached in the secondary tree through $[b, c]$ and b is reached in the secondary tree through $[a, b]$. Thus, b cannot be reached in the primary tree.

Figure 1.c shows two arc disjoint spanning trees. As stated earlier, any any redudant graph contains at least two such trees. Note that the failure of the edge indicated on the figure entails the failure of both arcs associated with that edge and thus vertex b becomes disconnected from the source.

Figure 3 shows two arc disjoint spanning trees using [1, 2, 3]. We call the trees Multicast Forward Backward (MFB) Trees. Here, failure of a link will not disconnect any node from s. However, the algorithms in [1, 2, 3] unnecessarily limit the tree choices. Figure 4 shows a pair of trees which the algorithm in [1, 2] and in this paper can achieve but which cannot be achieved by [3]. Figure 5 shows a pair of trees which the algorithm in this paper can achieve but which cannot be achieved with [1, 2]. Such choices may be more desirable for a variety of reasons, e.g. available bandwidth, transmission distance (as in this example), link costs, etc. We do not explicitly consider application dependent cost metrics in this paper, but rather focus our attention here on extending the possible multicast routing choices heretofore available, as illustrated in the above examples.

3 Automatic protection switching with trees.

Let us consider that we have a node redundant undirected graph $\mathcal{G}(\mathcal{N}, \mathcal{E})$. In the following, we shall use the terms "node" and "vertex" interchangeably. We shall say that the arcs (x_1, x_2) and (x_2, x_1) correspond to the undirected edge $[x_1, x_2]$. We wish to show that, for any source node $s \in \mathcal{N}$, we may create two directed trees, which we shall name B and R, for Blue and Red, such that, after eliminating any node (in the node redundant case) or any edge (in the edge redundant case), s remains connected to all nodes of $\mathcal{N}$ through B and/or through R deprived of the eliminated node or edge, respectively.

We start by choosing a cycle containing s. If this cycle does not include all nodes in the graph, we then choose a path that starts on some node in the cycle, passes through some set of nodes not on the cycle, and ends on another node on the cycle. If the cycle and path above do not include all nodes of the graph, we again construct another path, starting on some node already included, passing through one or more nodes not included, and then ending on another already included node. The algorithm continues to add new nodes in this way until all nodes are included.

We must also assign directions on the paths. Assigning directions is best thought of in terms of an analogy. Put a voltage source in node s, and put positive resistors on each edge of the cycle and added paths (removing all other edges of the original graph from consideration). The directions are then the directions of current flow. The nodes are also ordered in terms of their voltage (node s is special, and has two voltages, one on each side of the battery). Any such set of directions will yield B and R trees with the desired properties, and the algorithm below simply chooses one of these possible sets of directions.

We start with a node redundant graph, $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ and a source node s. The algorithm chooses a cycle, and then subsequent paths, and also orders the node 'voltages' as they get included in the cycle or one of the paths. We associate two arbitrary 'voltages' with s, namely $v^B(s)$ and $v^R(s)$, where $v^B(s) > v^R(s)$. For simplicity, we also sometimes refer to $v^B(s)$ as $v(s)$ in the algorithm. The set of nodes which have already been assigned a voltage at stage j of the algorithm is denoted by $\mathcal{N}_j$. At each stage j of the algorithm, a Blue (B_j) tree, $(\mathcal{N}_j, \mathcal{E}_j^B)$ and an Red (R_j) tree $(\mathcal{N}_j, \mathcal{E}_j^R)$ are formed, both of which span the set of nodes $\mathcal{N}_j$.

ALGORITHM FOR THE NODE REDUNDANT CASE

1) Set $j = 1$

2) Choose any cycle $(s, c_1, \dots, c_k, s)$ in the graph with $k \geq 2$. Let $\mathcal{N}_1$ be the set of nodes $\{s, c_1, \dots, c_k\}$ and order these nodes by $v^B(s) > v(c_1) > \cdots > v(c_k) > v^R(s)$

3)
$$\mathcal{E}_1^B = \{(s, c_1), (c_1, c_2), \dots, (c_{k-1}, c_k)\}$$
$$\mathcal{E}_1^R = \{(s, c_k), (c_k, c_{k-1}), \dots, (c_2, c_1)\}$$

4) If $\mathcal{N}_j = \mathcal{N}$, then set $B = B_j$, $R = R_j$ and terminate

5) $j := j + 1$

6) Choose a path $\left(x_{j,0}, x_{j,1}, \dots, x_{j,L_j}\right), L_j \geq 2$, in the graph such that $x_{j,0} \in \mathcal{N}_{j-1}$, $x_{j,L_j} \in \mathcal{N}_{j-1}$, with $v(x_{j,0}) > v(x_{j,L_j})$. The other nodes, $x_{j,i}, 1 \leq i < L_j$ are chosen outside of $\mathcal{N}_{j-1}$

7) $\mathcal{N}_j = \mathcal{N}_{j-1} \bigcup \{x_{j,1}, \dots, x_{j,L_j-1}\}$

8) Order the new nodes by $v(x_{j,0}) > v(x_{j,1}) > \cdots > v(x_{j,L_j-1}) > v_{max}$ where

$$v_{max} = \max \left[v^R(s), \max_{y \in \mathcal{N}_{j-1}} (v(y) : v(y) < v(x_{j,0})) \right]$$

9)
$$\mathcal{E}_j^B = \mathcal{E}_{j-1}^B \bigcup$$
$$\left\{ [x_{j,0}, x_{j,1}], [x_{j,1}, x_{j,2}], \dots, [x_{j,L_j-2}, x_{j,L_j-1}] \right\}$$
$$\mathcal{E}_j^R = \mathcal{E}_{j-1}^R \bigcup$$
$$\left\{ [x_{j,L_j}, x_{j,L_j-1}], [x_{j,L_j-1}, x_{j,L_j-2}], \dots, [x_{j,2}, x_{j,1}] \right\}$$

10) Go to step 4

For the sake of brevity, we omit the proof of validity.

In the case of switch protection for link failure, the above algorithm for node failure protection does not work in this case, because it is not always possible at a stage $j \geq 2$ to find paths as above. Sometimes, it is necessary to find a cycle that leaves the set of nodes $\mathcal{N}_{j-1}$ on one node and returns to the same node, i.e. with $x_{j,0} = x_{j,L_j}$. This can be handled by letting each node x have two 'voltages,' $v^B(x)$ and $v^R(x)$ associated with it. The ordering in step 2 of the algorithm is then replaced with

$$v^B(s) > v^B(c_1) > v^R(c_1) > v^B(c_2) > v^R(c_2) > \cdots$$
$$> v^B(c_k) > v^R(c_k) > v^R(s).$$

The ordering in step 8 of the algorithm is replaced in a similar way.

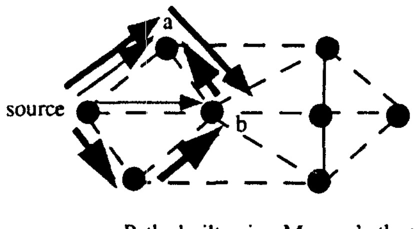
Finally, note that in the previous algorithm ([1, 2]), the selection of the paths or cycles in Step 6 was limited to start on a particular node.

4 Conclusions.

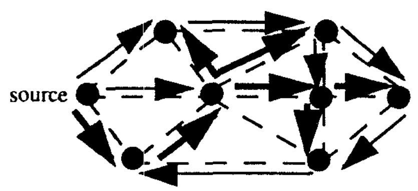
We have extended an algorithm which, on any node (edge) redundant network, forms two directed trees such that failure of any node (link) leaves the source connected to every unfailed node over at least one of the trees. Such trees are highly applicable for automated path protection for multicast and broadcast applications. The current algorithm produces the most general known construction for redundant trees over arbitrary redundant topologies and hence provides more freedom in network design and operation. In addition, the new algorithm is conceptually simpler than the previous approach.

Note that the above algorithm chooses the cycle or paths fairly arbitrarily but that various cost functions or constraints can be applied in the selection process. Thus, cost functions, such as transmission distance, congestion, and delay minimization, which are often associated with Steiner trees for multicasting can be taken into account when we select our trees. Of course, such considerations may increase the complexity of the algorithm.

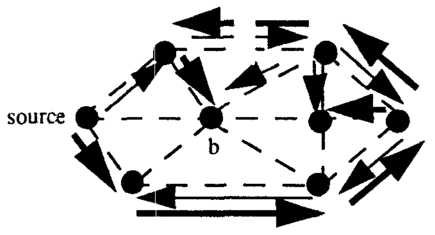
There are several other possibilities for further investigation of which the issue of capacity constraints is probably the most natural extension. Another extension is the construction of trees tolerant of multiple failures.



a. Paths built using Menger's theorem.



b. Edge-disjoint trees.



c. Arc-disjoint trees.

Figure 1: Three different approaches to building redundant trees. The dashed lines show the edges, the thin arrow lines show the primary tree and the thick arrow lines show the secondary tree. For figure a, the grey lines indicate one pair of edge-disjoint paths and the full lines indicate another pair of edge-disjoint paths. The interrupted lines correspond to failed edges.

References

[1] M. Médard, S.G.Finn, R.A. Barry, "Automatic Protection Switching for Multicasting in Optical Mesh Networks", in *Proceedings of OFC'97.*

[2] S.G. Finn, M. Médard, R.A. Barry, "A Novel Approach to Automatic Protection Switching Using Trees", in *Proceedings of ICC'97.*

[3] A. Itai, M. Rodeh, " The Multi-Tree Approach to Reliability in Distributed Networks", *Information and Computation*, vol. 79, 1988, pp. 43–59.

[4] T.-H. Wu, W.I. Way, "A Novel Passive Protected SONET Bidirectional Self-Healing Ring Architecture", *IEEE Journal of Lightwave Technology*, vol. 10, no. 9, September 1992.

[5] T.-H. Wu, *Fiber Network Service Survivability*, Artech House, 1992.

[6] J. Edmonds, "Edge-Disjoint Branchings", in *Combinatorial Algorithms, Courant Institute Computer Science Symposium 9*, January 24–25, 1972, Algorithmics Press, New York.

[7] M. Tomizawa, Y. Yamabayashi, N. Kawase, Y. Kobayashi, "Self-healing algorithm for logical mesh connection on ring networks", *Electronics Letters,* 15th September 1994, vol. 30, no. 19, pp. 1615-1616.

[8] D. Edinger, P. Duthie, G.R. Prabhakara, "A new answer to fiber protection", *Telephony*, April 9, 1990, pp. 53–55.

[9] R. Bhandari, "Optimal Diverse Routing in Telecommunication Fiber Networks", in *IEEE INFOCOM'94*, vol. 3, pp. 11c.3.1–11.c.3.11.

[10] J.W. Suurballe, "Disjoint Paths in a Network", *Networks*, 1974, pp. 125–145.

[11] B. Yener, Y. Offek, M. Yung, "Design and Performance of Convergence Routing on Multiple Spanning Trees", in *Proceedings of GLOBECOM'94*, vol. 1, pp. 169–175.

[12] K. Menger, "Zur allgemeinen Kurventheorie", *Fundamenta Mathematicae*, 10:96–115, 1927.

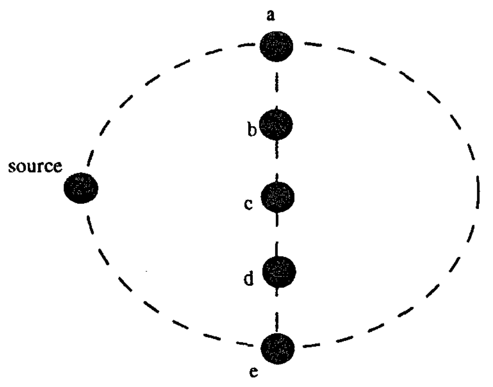


Figure 2: An example of a network which cannot have edge-disjoint trees.

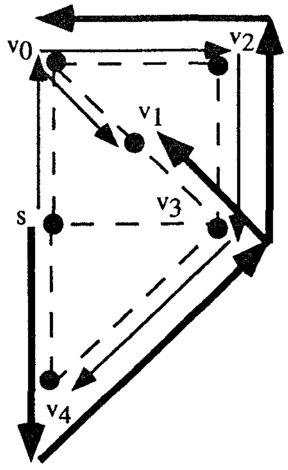


Figure 4: Trees which can be built by our algorithms but not by [3].

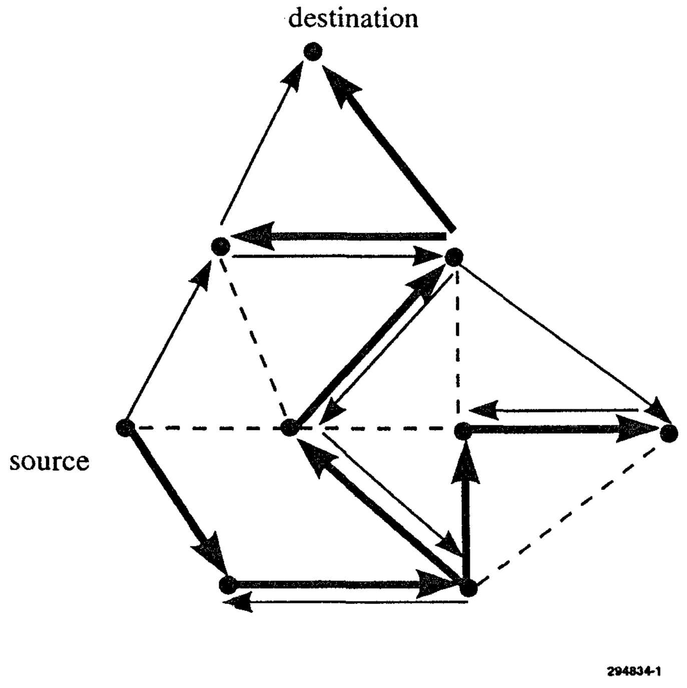


Figure 3: Application of our algorithm for an edge-redundant graph.

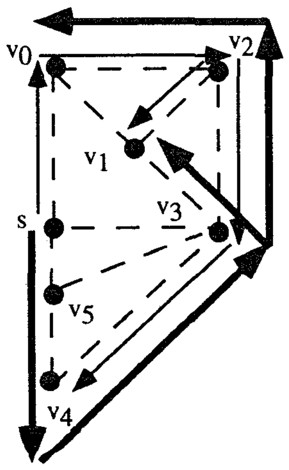


Figure 5: Trees which can be built by the algorithm in this paper but not by [1, 2].