Net Solver: A Software Tool For the Design of Survivable Networks*

Linda Morales Gardner, I. Hal Sudborough, and Ioannis G. Tollis

Department of Computer Science
The University of Texas at Dallas
Richardson, Texas 75083-0688

Abstract — Recent major interruptions of service in large metropolitan areas have demonstrated a need to design survivable networks. Of course, these networks must be of low cost to be practical. We present a software tool, called *Net Solver*, that designs survivable networks based on the self-healing ring architecture. Our tool iteratively finds a good ring cover and routing for all point-to-point traffic on a given network. The solutions produced by *Net Solver* are 10-50% better than those initially chosen or than those produced by hand. We have implemented a prototype that runs on a PC. The solutions are obtained in seconds or minutes, and include the rings, link capacities, routing and global cost.

1. Introduction

Fault tolerance in telecommunication networks is essential, especially in modern business and defense applications. Recent major interruptions of service in large metropolitan areas have demonstrated a need to design survivable networks. Of course, these networks must be of low cost to be practical. The basic problem is that a cable may be cut or broken due to ground digging by construction crews, earthquakes, or even sabotage. A network is defined to be *survivable* if it can sustain the loss of any single link. There has been considerable previous work on survivability in networks, see, for example [1, 2, 3, 7, 8, 11, 13, 14, 15, 16]. Ring architectures have received considerable attention recently [1, 3, 8, 11, 13, 14, 15], particularly with the advent of the *Synchronous Optical Network* (SONET) standard. In this paper we present a software tool, called *Net Solver*. Given an underlying network and a traffic matrix, *Net Solver* computes a cost-efficient, survivable network, based on a ring cover of all nodes, including a routing for all traffic on these rings. Our previous tool [1] described optimal ring covers for existing networks and a pre-assigned routing of traffic, in which every network link needed to be part of a ring. In contrast, *Net Solver* actually calculates the routing of traffic and does not include every network link in a ring. Instead, our requirement is that every node is part of a ring.

The input to *Net Solver* is a description of an existin network, a table of the point-to-point traffic, and the cos information for the ring technologies to be employed. It i important to note that networks are not designed fron scratch, but are designed on an existing underlyin connection structure. *Net Solver* identifies a ring cover for th given network and a routing of all traffic on the rings of th cover. Currently *Net Solver* computes ring covers using uni directional rings, four-fiber bi-directional rings, an combinations of these. (The user selects which technolog and associated costs the tool is to use.)

Let G=(V,E) denote a network, where V is a finite set c *nodes* and E is a set of undirected *links*. A node describes central switching point in the network and a link {x,y describes a connection between nodes x and y. A link {x,y is said to be *incident* to the two nodes x and y. A *ring* R is set of links that forms a circle, *i.e.* a set whose links can forr a path from a node x back again to x passing no (other) noc twice. A ring R *covers* a node x if there are links in R whic are incident to x. A *ring cover* *C* is a set of rings that cove all the nodes in the network [1, 10].

Let *C* be a ring cover and T a table describing, for a pairs of nodes in G, the amount of traffic (in DS3's) passin between the pair. *Net Solver* routes all the traffic in T on th links of rings in the ring cover *C*. Moreover, it finds a near optimal routing on these links, *i.e.*, a collection of routes fc all the traffic described in T that comes reasonably close t minimizing the cost for equipment. The user of *Net Solv* selects one of a given set of heuristics for routing. Th equipment cost for this routing is determined according to pre-assigned cost function provided by the user. *Net Solv* then considers alternative ring covers and routings using a iterative successive approximation technique. At each stage has a candidate ring cover *C*. It routes all the point-to-poi traffic on the links of *C* in an economical way and comput the cost of equipment and cable. This is denoted by cost(*C* Then, simple operations are used to transform *C* into ne candidate ring covers *C'*. The set of all ring covers obtain from *C* by these simple operations is called the *neighborho of C*. For each ring cover *C'* in the neighborhood of *C*, N

* This work was supported in part by a grant from Alcatel Network Systems.

GlobeCom '95

Solver computes cost(C'). When, for all ring covers C' in the neighborhood of C, cost(C') is larger than cost(C), *Net Solver* stops and announces the optimality of the ring cover C and the current routing. Otherwise, if there is a cheaper ring cover in the neighborhood of C, *Net Solver* chooses the most economical one, makes that the new current ring cover, and continues its search by looking at that ring cover's neighborhood.

Typically, *Net Solver* produces solutions 10-50% better than those initially chosen. For an example fifteen node network and traffic table the initial cost was roughly $8 million. The cost of the solution found by *Net Solver* is roughly $4 million. We do not claim that *Net Solver* always finds the best solution. The basic problem is, in fact, provably too difficult for global optimality.

2. Routing the Traffic

Our two principal routing heuristics include a feature common to all realistic formulas for the cost of equipment. The equipment needed on a node (or link) of the network depends upon the amount of traffic flowing through that node (or link). Moreover, the equipment cost for uni-directional and bi-directional rings grows in steps, so that for every additional d DS3's there is an additional cost of k, for some suitable constant k. This corresponds to the fact that current equipment is constructed to handle a specified number, say d, of DS3's. If the traffic is larger than d, then additional units of the equipment need to be purchased and each additional piece of equipment handles the same d DS3's of traffic. The bi-directional model used here assumes a cost function based on four-fiber cable. In fact, it should be noted that *Net Solver* works with any cost function. So, as technology changes, the program will still find an appropriate solution.

Net Solver currently employs two routing heuristics: *fewest hops* and *least resistance*. Routing by *fewest hops* is accomplished, for traffic between any pair of nodes {x,y}, by choosing a minimum length sequence of links from x to y. It should be noted that all links used in this routing must be part of some ring in the current ring cover C. (This is true for either *fewest hops* or *least resistance* routing.)

For the *least resistance* heuristic, consider traffic between a pair of nodes {x,y}, say of p DS3's. The *least resistance* heuristic chooses a path from x to y of minimum cost, but the cost of a particular link e on a possible path between x and y, when e is in a ring R, is defined to be the cost of extra equipment, if any, that must be added to all nodes in R to enable it to handle p additional DS3's. For example, if a four-fiber bi-directional ring R currently has equipment for handling q DS3's, but the actual maximum traffic on any link in R, even when a link in R fails, is m DS3's, with m<q, then there is extra, unassigned capacity in R. *Net Solver* exploits this potential for extra capacity in the second rou heuristic. Let equipment(R) denote the number of DS3's equipment on ring R can handle. For each link e, traffic(e), denote the number of DS3's so far routed on e. a ring R, equipment(R) is simply max(R), where max denotes the maximum value of traffic(e), (actually, smallest multiple of d larger than the maximum value traffic(e)), taken over all links e in R. The extra capacity c link e, denoted by extra(e), in a ring R, is the differen equipment(R) - traffic(e).

Observe that each link in a ring may have extra capaci as equipment typically needs to be provided in steps. That typically there is more capability for traffic in the curre equipment than will be needed, even when a link fails. follows that the cost of a link e for routing traffic betwe points x and y may be zero. That is, if there are p DS3's to routed and link e has extra capacity more than p, then t cost for link e is zero. More generally, if link e in ring R do not have sufficient extra capacity, equipment must be add when routing the stated traffic on link e (between x and y The amount of basic equipment needed is the amount that ca handle the number of DS3's represented by (p - extra(e multiplied by the number of nodes in R. The multiplication needed, as every node in the ring will need the sam additional equipment. This computation gives the amount c basic node equipment required over and above the equipmer already allocated to the ring. It should be noted that the actua cost of the network also includes the cost of tributary shelve and other items needed at source and destination nodes and a nodes where traffic switches between rings.

Least resistance routing finds the most economical rout for the traffic between points x and y when the cost of using an edge is computed by the formula indicated above. Observe that *least resistance* routing can be quite different from *fewest hops* routing. For example, it may be that points x and y are connected by a single link, in which case *fewest hops* routing will always route traffic between x and y on this link. However, it may be that to do so requires a large amount of additional equipment, and a more circuitous route exists, albeit with many additional links, but with zero additional equipment cost .

We note that in *least resistance* routing the route chosen for traffic between a pair of nodes is strongly dependent on the routes chosen earlier for other traffic. That is, routes chosen earlier may cause nodes along some path P to need no new equipment to handle some new traffic, as equipment already in place may be sufficient. On the other hand, it may be that previously chosen routes may fully saturate equipment at nodes along path P and hence another path would be less expensive. Thus, the suitability of a path P is often dependent on traffic previously routed.

Due to this dependence of current path choices (in *least resistance* routing) on routes previously chosen, it was decided to make *Net Solver* route traffic in order of diminishing size. That is, it chooses first a pair of nodes with the largest amount of traffic passing between them, routes it, next chooses a pair with the largest remaining amount of traffic, routes it, and so on, until all traffic has been routed. *Net Solver* also allows *two pass* routing. With *two pass* routing the traffic is routed twice. That is, traffic is re-routed with all other traffic present, and consequently, paths of least resistance may be quite different in the end.

It should be noted that routing algorithms must also consider cases when there are several links between the same pair of nodes. That is, a link in the given network may be part of several rings and *Net Solver* needs to decide which of the many rings connecting the pair of nodes is best. This is done in a manner similar to what was described above, by selecting the ring with the least total cost.

3. Finding Good Ring Covers

Net Solver also selects a ring cover *C*. As the number of possible ring covers is very large, *Net Solver* looks at a subset of all possible ring covers. *Net Solver* is given a ring cover C_0 as a starting point. It need not be a particularly good ring cover from the view of minimizing equipment costs. *Net Solver* then proceeds iteratively to find a nearly optimum ring cover *C* and a nearly optimum routing of all the traffic on the rings of *C*.

Net Solver works by investigating ring covers in a well defined *neighborhood* of its current ring cover *C*. The definition of neighborhood is given through a set of operations on rings, including:

- *splitting*
- *merging*
- *expanding*

That is, the *neighborhood* of a ring cover *C* is the set of all ring covers obtainable from *C* by these operations. The next few paragraphs describe the operations in detail. For the sake of these explanations, we assume rings are drawn in the plane, so we use notions of *clockwise* or *counterclockwise* motions in describing ring traversals.

The *splitting* operation splits a ring R into two rings, say R_1 and R_2, using a *chord* of R. That is, let {x,y} be a chord of R, as illustrated in Figure 1. The splitting operation would split the ring R into two rings, say R_1 and R_2, where R_1 is the ring defined by the edges on the path from x to y on the links of R, traveling in a clockwise direction, together with the link {x,y}, and R_2 is analogously defined by the edges on the path from x to y traveling in a counterclockwise direction. The two rings R_1 and R_2 are illustrated in Figure 1.

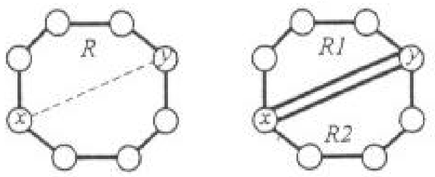

Figure 1: Split operation

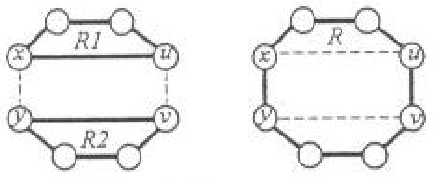

Figure 2: Merge operation

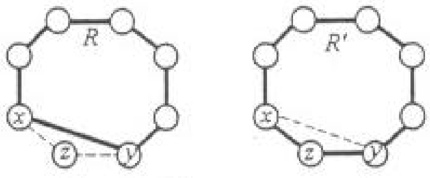

Figure 3: Enlargement operation

There are currently three *merge* operations. The first merges two rings that share a link. That is, if R_1 and R_2 are rings, there is a link {x,y} in both R_1 and R_2, and no other link common to both R_1 and R_2, then one forms a ring R containing all the links of R_1 and all the links of R_2, except for the common link {x,y}. So, if there are m links in R_1 and n links in R_2, there will be m+n-2 links in the ring R formed by merging R_1 and R_2. The second merge operation puts together two rings connected by a *bridge*. This is illustrated in Figure 2. A bridge is a pair of links, say e_1={u,v} and e_2={x,y}, such that (a) u and x are consecutive nodes in a clockwise traversal around R_1 and (b) v and y are consecutive nodes in a counterclockwise traversal around R_2. The new ring R consists of all links in R_1 and R_2, together with e_1 and e_2, but not including the link {u,x} in R_1 or the link {v,y} in R_2. The last merge operation puts together two rings that share exactly one node. That is, let R_1 and R_2 be rings, let x be a node included in both rings, and let y and z be neighbors of x on rings R_1 and R_2, respectively, such that {y,z} is a link. Then, one forms a new ring R by including all the links in R_1 and R_2, except for {x,y} and {x,z}, together with the link {y,z}.

The *expansion* operation makes a ring larger; it expands a ring R into a new ring R' that covers all nodes covered by R and one additional node. That is, let R be a ring, let {x,y} be a link in R, and z be a node not covered by R. If z is adjacent to both x and y, then R' is the ring with all the links in R, except the link {x,y}, together with the links {x,z} and {y,z}. This is illustrated in Figure 3.

Note that there may be many ring covers in the neighborhood of a given ring cover *C*. *Net Solver* needs to choose the best one. To do so it routes all point-to-point traffic using the chosen heuristic and computes the cost of equipment. The computed cost for a ring cover *C'* is denoted by cost(*C'*). The best ring cover is the one of least cost. Let *C'* be the best ring cover in the neighborhood of *C*. If cost(*C*) ≤cost(*C'*), *Net Solver* stops and announces the optimality of *C* and describes the routes for all traffic on the rings of *C*. If cost(*C'*) < cost(*C*), *Net Solver* continues its search with *C'* as the current ring cover and investigates the ring covers in the neighborhood of *C'*. It should be noted that *Net Solver's* search always terminates, as the cost is a natural number and must decrease with each iteration.

4. Experimental Results

We have run a number of experiments using several network topologies and equipment costs. These employed uni-directional and four-fiber bi-directional ring technologies, as well as hybrid mixtures of the two technologies. The results of some of our experiments are summarized in Table 1 and an example is shown in Figure 4. As indicated earlier, these results are obtained in minutes. Better results can be obtained, of course, by allowing for more computational effort.

Table 1. Sample results obtained by *Net Solver*.

Size of network	Ring Type	Initial Cost	Final Cost	Cost Reduction
15 nodes	bi-dir	$6,500,100	$4,170,960	36%
	uni-dir	$8,593,650	$4,560,070	47%
49 nodes, light traffic	bi-dir	$35,863,640	$23,925,080	23%
	uni-dir	$51,164,240	$25,462,520	50%
49 nodes heavy traffic	bi-dir	$67,255,080	$45,344,140	33%
	uni-dir	$89,249,200	$47,382,040	47%

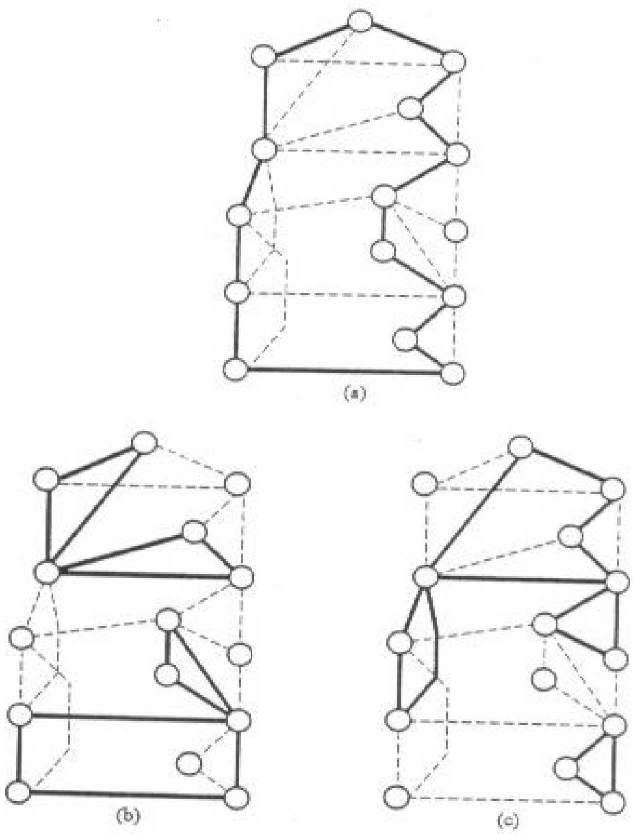

Figure 4. (a) A 15 node network and an initial ring cover, (b) and (c) together describe the rings in a ring cover obtained by *Net Solver*.

5. Conclusions and Future Work

Our experience with *Net Solver* has been very positive. Typically it produces solutions that are 10-50% better than those initially chosen, or than those produced by hand. With our search procedure, the solution may get stuck at a local minimum, which is significantly far away from the global minimum solution. We plan to investigate new techniques for solving this problem based on simulated annealing [12]. Simulated annealing techniques have produced very good results in several areas with similarly difficult problems, such as VLSI layout. Also, we plan to investigate alternatives to ring architectures, such as meshes, two (or more) link-disjoint spanning trees, etc. [4,5]. We plan to study better routing heuristics such as multicommodity flow techniques [6].

Acknowledgement

We would like to thank Jay C. Shah for useful discussions and comments.

References

L. M. Gardner, M. Heydari, J. Shah, I. H. Sudborough, I.G. Tollis, and C. Xia, Techniques for Finding Ring Covers in Survivable Networks, *Proc. 1994 IEEE GLOBECOM*, November 1994, pp. 1862-1866.

B. Gavish, P. Trudeau, M. Dror, M. Gendreau and L. Mason, Fiberoptic circuit network design under reliability constraints, *IEEE J. on Selected Areas in Communications*, Vol. 7, No. 8, October 1989, pp. 1181-1187.

W. D. Grover, B. D. Venables, J. H. Sandham, and A. F. Milne, Performance studies of a self-healing network protocol in Telecom Canada long haul networks, *IEEE GLOBECOM 1990*, pp. 403.3.1-403.3.7.

M. Grötschel, C. Monma and M. Stoer, Polyhedral approaches to network survivability, *DIMACS Series in Disc. Math. and Theoret. Comp. Sci.*, Vol. 5, 1991, pp. 121-141.

S. Khuller, B. Raghavachari and N. Young, Designing multi-commodity flow trees, *Proc. of 3rd Workshop on Algorithms and Data Structures*, (1993), LNCS 709, pp. 433-441.

F. T. Leighton, F. Makedon, S. Plotkin, C. Stein, É. Tardos and S. Tragoudas, Fast approximation algorithms for multicommodity flow problems, *Proc. of 23rd ACM Symp. on the Theory of Computing*, (1991), 101-111.

K. T. Newport and P. K. Varshney, Design of survivable communication networks under performance constraints, *IEEE Transactions on Reliability*, Vol. 40, No. 4, October 1991, pp. 433-440.

H. Sakauchi, Y. Nishimura and S. Hasegawa, A self-healing network with an economical spare-channel assignment, *IEEE GLOBECOM 1990*, pp. 403.1.1-403.1.6.

[9] J. C. Shah, DS3 restoration network design tool design specification, Manuscript, 1990.

[10] I. H. Sudborough, I. G. Tollis, L. M. Gardner, M. Heydari, and C. Xia, Advanced network topologies for network survivability, Technical Report submitted to Alcatel Network Systems, Inc., January 1993.

[11] O. J. Wasem, An algorithm for designing rings for survivable fiber networks, *IEEE Transactions on Reliability*, Vol. 40, No. 4, October 1991, pp. 428-432.

[12] D. F. Wong, H. W. Leong, C. L. Liu, Simulated annealing for VLSI design, Kluwer Academic Publishers, 1988.

[13] T. H. Wu, D. J. Kollar, and R. H. Cardwell, Survivable network architectures for broad-band fiber optic networks: model and performance comparisons, *IEEE Journal of Lightwave Technology*, Vol. 6, No. 11, November 1988, pp. 1698-1709.

[14] T. H. Wu, D. Kong and R. Lau, An economic feasibility study for a broadband virtual path SONET/ATM self-healing ring architecture, *IEEE J. Selected Areas in Communications*, Vol. 10, 1992, pp. 1459-1473.

[15] T. H. Wu and R. C. Lau, A class of self-healing ring architectures for SONET network applications, *IEEE GLOBECOM 1990*, pp. 403.2.1-403.2.8.

[16] T. Yokohira, M. Sugano, T. Nishida and H. Miyahara, Fault tolerant packet-switched network design and its sensitivity, *IEEE Transactions on Reliability*, Vol. 40, No. 4, October 1991, pp. 452-460.