Techniques for Finding Ring Covers in Survivable Networks[1]

L. M. Gardner[2], M. Heydari[3], J. Shah[4], I. H. Sudborough[2], I. G. Tollis[2], and C. Xia[2]

Abstract—Given an arbitrary telecommunications network N our goal is to find the minimum cost for equipment which will enable N to survive an arbitrary link fault. We consider uni-directional and bi-directional ring technologies. Basically, our goal is to find minimum cost *ring covers* for any network N, where a ring cover is a set C of rings such that every link in N is covered by (*i.e.* part of) at least one ring in C. If a network N is augmented with enough equipment to support a given ring cover C, it can respond to a link failure immediately (and automatically) by routing the disrupted traffic through surviving links in the ring that covers the failed link.

We describe an efficient algorithm to find a minimum cost ring cover for uni-directional transmission rings under simplifying assumptions. This algorithm offers a useful heuristic for computing low cost ring covers for existing networks and actual cost functions. We also provide efficient heuristics to find nearly minimum cost ring covers for bi-directional transmission rings. We show that certain versions of the bi-directional problem are *NP*-complete, hence (presumably) no efficient algorithm exists that *always* finds a minimum cost ring cover. However, our heuristics perform well in practice.

1. Introduction

Network survivability is critical. For instance, a recent failure of a telecommunications network led to the isolation of New York City and the shutdown of JFK airport. The failure of a fiber link not only disrupts the communication normally passing through it, but may disrupt the entire network as problem escalation through unplanned circumventions of the broken link are attempted. The ultimate goal is to provide self-healing capabilities to a network. That is, the goal is to design networks that can survive equipment outages, link cuts, and even natural disasters such as fire, floods, and earthquakes. The first step is to compute the cost of augmenting an existing network with the minimum amount of equipment needed to survive an arbitrary link cut. A multi-ring architecture is a common approach. Ring architectures have received considerable attention, particularly with the advent of the *Synchronous Optical Network* (SONET) standard [1-6].

Let G=(V,E) denote a graph, where V is a finite set of *nodes* and E is a set of undirected *links*. A *network* N consists of a connected graph G together with a set W of positive integer weights associated with each link in E. A node describes a 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 nodes x and y. A *ring* R is a set of links that forms a circle, *i.e.* a set whose links form a path from a vertex x back again to x passing no other node twice. A ring R *covers* a link *e*, if *e* is in R. A *ring cover* C is a set of rings that cover all links in the network.

Let C be a ring cover and T a table, describing for all pairs of nodes in G, the amount of traffic (in DS3's) passing between the pair. The amount of equipment needed on a node (or link) of the network depends upon the amount of traffic flowing through that node (or link). Moreover, equipment costs grow in steps, so that for every additional *d* DS3's there is an additional *kd* cost, for a 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 equipment need to be purchased; each additional piece of equipment handles the same *d* DS3's of traffic.

Our goal is to find a ring cover C for N that necessitates equipment of minimum cost. This problem is simply stated, but difficult computationally. We note that a network N with *m* links may have 2^m different rings and, hence, a doubly exponential number (in *m*) different subsets of rings. Each is a candidate for best ring cover. Although the typical number of candidate ring covers prohibits computing all and sequentially determining which is best, there may be cases where the network is either very small or can be logically decomposed into separate small pieces to be covered separately. In these cases an algorithm to list all rings, and hence also all possible ring covers, may be useful. We describe here briefly such an algorithm.

Let N be a network for which the set of all possible rings is to be computed. Begin with a *depth first search* routine [7,8]. Depth first search starts from a specified node, proceeds through all links of N, and forms a *depth first spanning tree* of N. (A spanning tree is a tree containing all nodes in the network.) When a link *e* is traversed, during depth first search, that returns, say from a node *y* to a node *x* previously visited, a *basic cycle* is formed. That is, in this case, the basic cycle consists of the links in the spanning tree from *x* to *y* together with link *e*. The collection C of all basic cycles is called a *basis* for the desired set of all cycles. (A basis set for a vector space is an appropriate analogy.) In fact, the desired set S of all rings is the set of all linear combinations of *basic cycles*, with the operation being the *exclusive or* (or set difference) operation. We have written such a program to list all possible rings for a network N, but (as mentioned) its

[1]This research was funded by a grant from Alcatel Network Systems, Inc., Richardson, Texas.
[2]Computer Science Program, Jonsson School of Engineering and Computer Science, University of Texas at Dallas, Richardson, Texas, 75083-0688.
[3]Department of Mathematics and Computer Science, University of Wisconsin-Whitewater, Whitewater, WI 53190.
[4]Alcatel Network Systems, Inc. Richardson, Texas.

GlobeCom '94

feasibility is limited. For example, for a complete network of six nodes (*i.e.* a network with connections between every pair of nodes), the number of different rings is 197 and, for a complete network of seven nodes, the number of different rings is over one thousand.

2. Ring Covers for Uni-Directional Ring Technologies

In the case of a uni-directional ring R, the cost of equipment for R is directly proportional to the sum of DS3's for traffic routed on R, where the sum is taken over all links in R. Although a typical equipment cost function is described by a *step* function, e.g. more equipment is required for each additional 24 DS3's, our initial simplifying assumption is to regard the cost as a linear function of traffic. Our procedure, called *Eulerian Ring Cover*, computes the least expensive ring cover under such simplifying assumptions and can be used, therefore, as a means of computing excellent (but perhaps not optimum) ring covers for existing networks and actual equipment costs. We note that the cost of a ring cover C for the entire network N is the sum of costs for each ring in C.

We assume we have no visibility of individual source-destination traffic, but traffic has been routed already on a link-by-link basis. Thus, each link has associated with it an integer describing how much traffic (in *DS3's*) it may need to handle. We shall call the integer associated with a link e the *weight* of that link. Our goal is to find the minimum cost ring cover for N with the indicated weights. As indicated, the cost of a ring cover C is directly proportional to the sum of weights on all links of rings in C, assuming a worst case of traffic distribution. Our simplification takes the cost of a ring to be precisely this sum. Thus, we have:

Min-Sum Ring Cover (MSRC) *Problem:*
Input: A network N.
Output: A ring cover C of minimum cost, where the cost of a ring cover C is the sum, over all rings R in C, of the sum of weights of links in ring R.

We have programmed an efficient algorithm to solve the MSRC problem. The rest of this section describes our procedure.

A network N is said to be *Eulerian* [10] if there exists a single *cycle* that covers every link. (This ring is called an *Eulerian cycle*. Note that a *cycle* differs from a *ring*. That is, a cycle is a circular path in the network, as is a ring, but following the path described by a cycle may mean visiting a node more than once.) Euler proved that a network is *Eulerian* if and only if every node is incident to an even number of links [9].

An *Eulerian cycle* K can be decomposed into a ring cover. That is, suppose one starts from an arbitrary node and follow K's links until a node is visited for a second time. This portion of K's path describes a ring R. Delete all links in R. The remaining network is still Eulerian (as every node is incident to an even number of links). In fact, the remainder of K is an *Eulerian cycle* for this network. So, one can repeat the previous step and continue to obtain rings until all links in K have been exhausted.

If a network is *Eulerian*, then a minimum cost (hence an optimum) ring cover is obtained for it by decomposing an *Eulerian cycle* K into a ring cover C. The decomposition is obtained by following links in the *Eulerian cycle* K until a node is revisited. This portion then forms a ring and is added to the ring cover C. That the decomposition produces an optimum ring cover follows easily from the fact that no two rings in C have a common link. That is, ring cover C requires the least expensive equipment, as any ring cover must cover all of the links in the network and the rings of C cover all links without overlap. Furthermore, as the cost of a ring R is the sum of the weights on all its links, and the cost of a ring cover C is the sum, over each of its rings R, of the cost of R, the least expensive ring cover is one in which rings do not share an edge. Thus, a ring cover C, obtained by decomposing an *Eulerian cycle* K, is a least expensive ring cover for an *Eulerian* network.

If a network N is not *Eulerian*, then it has some number of nodes incident to an odd number of links. (We refer to these nodes as nodes of *odd degree*.) In fact, N must have an even number of nodes of odd degree, as each link connects two nodes and therefore the total number of node incidences for the network must be even. Thus, one can augment N by adding links between pairs of nodes of odd degree and obtain a new *Eulerian* network N'. It should be noted that when one augments a network N by adding new links to connect pairs of odd degree nodes, the new links are routed along paths of already existing links. That is, no new right-of-way (*i.e.* land) purchases are required. One needs to (at worst) purchase additional cable and equipment to handle the new traffic. In fact, new links are not really added in a strict sense, they are only added conceptually as a device to achieve an *Eulerian* network whose *Eulerian cycle* can be decomposed into a ring cover. Moreover, the links added in augmenting the network are routed along a path of minimum total weight, so that the cost of augmentation is minimized. (This is necessary in order to deduce the optimality of our procedure.) Note that any routing for the added links does not change the parity of nodes routed through. That is, if a node x has even degree and an added link is routed through x, then the number of links incident upon x is increased by two and hence x still has even degree. So, after pairs of odd degree nodes are connected by augmenting links and the new links are routed, the network does indeed become *Eulerian*.

Which pairs of odd degree nodes should be connected in order to construct an augmentation allowing an *Eulerian cycle* K of minimum total weight? The answer is: Those pairs described by a *minimum weight matching* [7,8] on nodes of odd degree. That is, let G'=(V',E') be such that V' is the set of all nodes of odd degree in N and E' is all possible edges between nodes in V' (*i.e.* G' is a complete graph on the odd degree nodes of N). Let an edge {x,y} in G' have weight *w*, where *w* is the weight of the minimum weight path between x and y in N. That is, let P be a path between x and y in N and let P consist of the links $e_1, e_2, ..., e_k$. The weight of P is the sum of the weights on $e_1, e_2, ..., e_k$. The weight assigned to the edge {x,y} is the minimum weight of a path P, taken over all paths P between x and y. The desired minimum weight matching is a complete matching of the nodes in G'.

Let G be a graph with weights assigned to its edges. A *matching* in G is a set of edges M such that no two edges in M are incident to the same vertex. A matching M is *perfect* if every node in G is incident to an edge in M. The *weight* of a matching M, denoted by *weight(M)*, is the sum of all weights assigned to edges in M. A *minimum weight perfect matching* is a perfect matching M such that, for all perfect matchings M' in G, weight(M) ≤ weight(M').

There are efficient algorithms described in the literature (see, for example, [7,8]) that compute a minimum weight perfect matching of a complete edge weighted graph. For example, such a matching can be obtained in time bounded by O(m n log n), where n is the number of nodes and m is the number of edges [7,8]. (Let n be the number of odd degree nodes in the network. Then, as G has every possible edge between odd degree nodes, the O(m n log n) time bound can be written as $O(n^3 \log n)$.) We have implemented such an algorithm and use it as a principal component in an algorithm to compute a minimum cost ring cover for a network.

The algorithm *Eulerian Ring Cover* computes a minimum cost ring cover for a network N as follows:

- It creates the complete graph G' on the odd degree nodes of N and assigns each edge {x,y} of G' the weight of the minimum weight path between x and y in N. (The weight of a minimum weight path between x and y is obtained by a shortest path algorithm [7,8].)
- It obtains a minimum weight perfect matching M for G' and uses the matching to augment the network N into an *Eulerian* network N'. The augmentation adds links between pairs of nodes, say {x,y}, described in the matching M and routes these links along paths of minimum weight between x and y.
- It constructs an *Eulerian cycle* K in the *Eulerian* network N'.
- It decomposes the *Eulerian cycle* K into a ring cover C.

There is a subtle issue hidden in the above procedure. When one decomposes an *Eulerian cycle* K into a ring cover C, one needs to be able to guarantee that each ring created is actually a ring in the original network N. It is possible that a ring in the augmented network N' might consist of a link between x and y together with an augmented link also between x and y. From the point of view of constructing a network capable of surviving any link fault, this will not do. That is, the original link between x and y and the augmenting link between x and y are routed over the same right-of-way; hence whatever occurs causing a break will presumably break both. Consequently, one wants to guarantee that the decomposition of the *Eulerian cycle* K into a ring cover C does not allow such rings to be included. This is not always possible, but *is* possible most of the time and is possible for networks satisfying certain specified conditions. For example, for any network N whose nodes can be covered by two edge disjoint spanning trees, one can always guarantee an appropriate decomposition of an *Eulerian cycle* [9]. The strategy to obtain an appropriate decomposition of an *Eulerian cycle* (under this condition) is straightforward and does not add to the overall time complexity of our procedure.

As the time for obtaining the minimum weight perfect matching dominates the time of the entire procedure, we arrive at the following theorem:

Theorem 1. The MSRC problem can be solved in $O(n^3 \log n)$ time, for networks N whose nodes can be covered by two edge disjoint spanning trees.

There are other conditions that are sufficient to guarantee an appropriate decomposition of an *Eulerian* cycle K into a ring cover C. These are described in [11]. As an illustration of our procedure, we include Figures 1-3. They show, respectively, a ten node network N with link weights, the *Eulerian* network N' obtained by adding links through a minimum cost matching of odd degree nodes, and a resulting ring cover C obtained by decomposing an *Eulerian cycle* of N'.

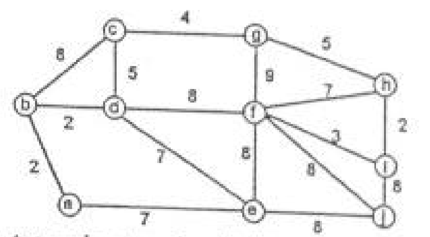


Figure 1. A ten node network with link weights corresponding to traffic.

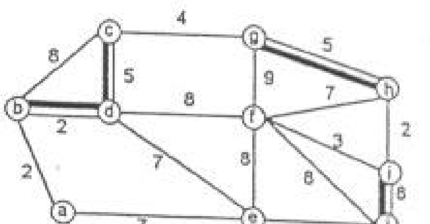


Figure 2. An Eulerian network obtained by a min cost matching of odd degree nodes (augmenting links are shown with bold lines).

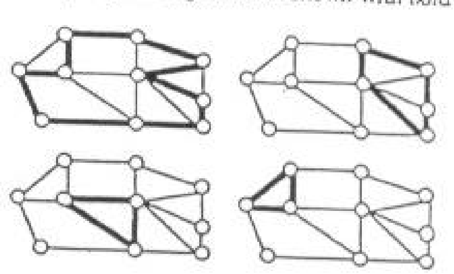


Figure 3. The ring cover obtained by our *Eulerian Ring Cover* procedure. The cover has four rings, shown with bold lines.

We note that, while our procedure finds an optimum ring cover, it is optimum for a problem created by several simplifying assumptions. In practice, therefore, this procedure will find utility as an excellent heuristic for obtaining economical covers with unidirectional rings.

3. Ring Covers with Bi-Directional Ring Technologies

When one employs bi-directional ring technologies, the cost for equipment on a ring is roughly proportional to the maximum amount of traffic passing through any link in the ring. That is, additional equipment needs to be placed around he links in a ring to handle traffic normally passing through a broken link in the ring. Clearly, the worst case is repre- ented by the link with the maximum amount of traffic. Hence, the cost of a ring is proportional to the maximum veight of any link in the ring. Thus, we have the following ing cover problem:

Iin-Max Ring Cover (MMRC) ***Problem:***
nput: A network N.
)utput: A ring cover C of minimum cost, where the cost of a ring cover C is the sum, over all rings R in C, of the maximum weight of a link in ring R.

We observe first that the MMRC problem is *NP-* ard. In fact, we show that the problem of determining, for a ven integer k, if there is a ring cover of cost k is *NP-com- ete*. The transformation is from the *undirected Hamiltonian rcuit* problem [12], *i.e.* the problem of whether a finite un- ected graph has a single ring that includes every node.

Let G=(V,E) be an arbitrary graph. Construct a new aph G'=(V',E'), where $V'=V_H \cup V_L$ and $E'=E_H \cup E_L$, where:
$V_H=\{x_1,x_2 \mid x \in V\}$ and $V_L=\{x_e,y_e \mid e \in V\}$, and
$E_H=\{\{x_1,x_2\} \mid x \in V\}$ and
$E_L=\{\{x_1,y_2\},\{y_1,x_2\},\{x_1,x_e\},\{x_e,y_2\},\{y_1,y_e\},\{y_e,x_2\} \mid e=\{x,y\} \in E\}$.

e nodes in V_H are called *heavy* nodes and the nodes in V_L called *light* nodes. Similarly, the edges in E_H are called *vy* edges and the edges in E_L are called *light* edges. Let m an integer, sufficiently large so that m^2 is much larger than , where m is the number of nodes in G. Then, assign the ght m^2 to all edges in E_H and assign weight one to all es in E_L. (Observe that the edges in E_L form two disjoint ngles, namely formed by the three nodes: x_1,y_2,x_e and the e nodes: y_1,x_2,y_e. Furthermore, one can view the con- cted graph G' as consisting of an edge of weight m^2 for node in G and a triangle connecting one end of a weight edge, say corresponding to a node x in G, to the opposite of a weight m^2 edge, say corresponding to a node y in G, n there is a link {x,y} in G.)

We claim that the constructed graph G' has a ring r of weight at most m^2+2m if and only if G has a *Hamil- an* circuit. One direction is straightforward. If G has a *iltonian* circuit, say one represented by the sequence of nodes x(1), x(2), ..., x(n), where x(1)=x(n), then one can form a ring in G' containing all of the edges with weight m^2. This ring consists of the nodes $x(1)_1$, $x(1)_2$, $x(2)_1$, $x(2)_2$, ..., $x(n-1)_1$, $x(n-1)_2$, $x(1)_1$. All other edges of G' can be included in rings of weight one. Thus, one obtains a ring cover of weight at most m^2+2m, as there are 2m triangles in G' (*i.e.* one for each edge of G).

Next suppose that there is a ring cover of weight at most m^2+2m for G'. This can only occur if there is a single ring that contains all of the edges with weight m^2; otherwise, there would be at least two rings in the cover with weight m^2 edges and hence the cover would have total weight at least $2m^2 > m^2$+2m. This unique ring that contains all the heavy edges of G' must pass through all of the heavy nodes once and, as the triangles of G' correspond to edges of G, it also describes a *Hamiltonian* circuit of G.

So, if there were a polynomial time algorithm to solve the *MMRC* problem, then there would be a polynomial time algorithm to solve all problems in the class *NP*. As this seems quite unlikely, we believe the *MMRC* problem to be intractable. Consequently, we have programmed various heuristics. The remainder of this section is devoted to a description of these heuristics and their performance on test data.

Basically all our heuristics use *depth first search* [7,8] to locate rings in the network. The heuristics differ in their manner of deciding which rings to include in the cover. In fact, they look for rings iteratively. That is, they keep a list of links yet to be included in a ring; which is ordered by some criterion. For example, links may be ordered by decreasing weight and a heuristic may choose rings by the rule: cover heaviest links first. A heuristic might proceed by depth first search from an endpoint of the first link on the list and terminate when it reaches the other endpoint of this first link, provided the object found really is a ring, *i.e.* contains at least three links. (We also require that a ring contain no more than sixteen links, as required by current technological constraints.)

We also implemented a procedure to choose where to begin depth first search. That is, a *priority* is assigned to each node and the node with highest priority is the point at which the search begins. After finding a ring that includes a node of highest priority, the procedure recomputes the priority of every node.

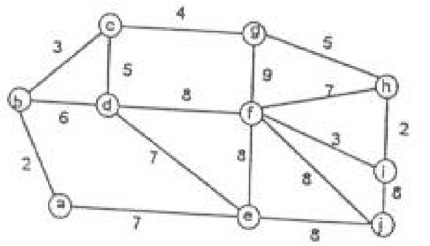
The following heuristics have been implemented:

- A *Greedy* heuristic that starts the search at the node of highest priority and takes the first ring found. This continues until all links are covered by some ring in the cover.
- A *Longest Feasible Ring* heuristic. This starts the search at the node of highest priority, but does not take the first ring found. Instead, it saves the information about each ring found and takes the one with the maximum number of links within the sixteen link constraint on ring size. It continues the same process until all links are covered by some ring in the cover.

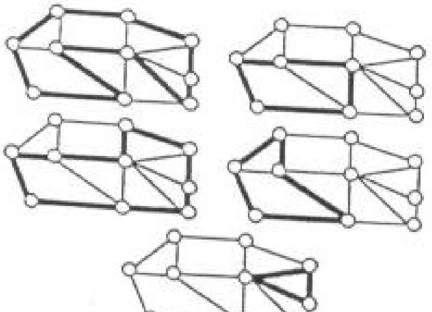
- A *Maximally Separated Rings* heuristic. This starts the search at the node of highest priority, but always takes a link not yet included in any ring in the cover when it is possible to do so. That is, this heuristic tries to avoid covering a link with many different rings. The procedure continues until all links are covered by some ring in the cover.

We tested our heuristics on several example networks vith weighted links. The last two heuristics consistently per- orm better than the first, but there is no apparent way to de- ide which of the last two heuristics is best. This seems to de- end upon the instance of *MMRC* chosen. Part of the prob- :m is a lack of sufficiently powerful analytical tools to etermine, for arbitrarily large networks, if one has an ptimum ring cover. Specifically, how does one determine ow far from optimum a candidate solution is? Bipartition sults can be used to obtain a rather crude lower bound. That , one divides the nodes into two disjoint sets, determines the affic passing between the two sets, and deduces a lower und on the weight of at least one ring. We shall not go rther here, but (with suitable extensions) this technique ight be useful.

We conclude with some illustrations indicating the e of results obtained on a ten node network. Specifically, consider the ten node network illustrated in Figure 4. Ring vers are shown with bold lines in Figures 5, 6 and 7.



ıre 4. A ten node network with link weights corresponding to ic.



'e 5. Ring cover obtained by the *Greedy* heuristic (cost = 39).

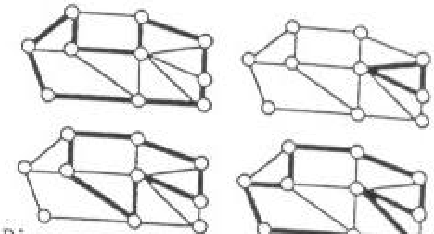


Figure 6. Ring cover obtained by *Longest Feasible Ring* heuristic (cost = 32).

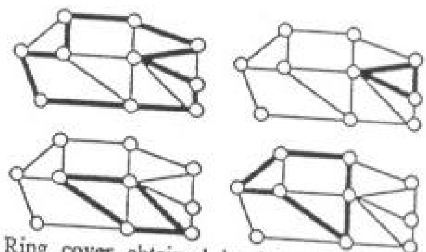


Figure 7. Ring cover obtained by *Maximally Separated Rings* heuristic (cost = 32).

References

[1] W. D. Grover, B. D. Venables, J. Sandham, and A. Milne, "Performance of the Self-healing Network Protocol with Random Individual Link Failure Time," *Proc. of IC 1991*, pp. 660-666.

[2] H. Sakauchi, Y. Nishimura, and S. Hasegawa, "A Self-Healing Network with an Economical Spare Channel Assignment," *Proceedings of IEEE Globecom '90*, pp. 444-452.

[3] T. Wu and C. Lau, "A Class of Self-Healing Ring Architectures for SONET Network Applications," *Proceedings of IEEE Globecom '90*, pp. 444-452.

[4] T. Wu, D. J. Kolar, and R. H. Cardwell, "Survivable Network Architectures for Broadband Fiber Optic Networks: Model and Performance Comparisons," *IEEE Journal of Lightwave Technology*, vol 6, no. 11, (Nov., 1988), pp. 1698-1709.

[5] J. C. Shah, "DS3 Restoration Network Design Tool Specification," *Manuscript*, Alcatel Network Systems, Richardson, Texas, 1990.

[6] J. C. Shah, "Restoration Network Planning Tool," *8th Annual Fiber Optic Engineers Conference*, April, 1992, held in Washington, D.C.

[7] B. M. E. Moret and H. D. Shapiro, *Algorithms from P to NP*, The Benjamin/Cummings Publishing Company, Inc., 1991.

[8] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, McGraw-Hill Publishers, 1990.

[9] A. Gibbons, *Algorithmic Graph Theory*, Cambridge University Press, 1985.

[10] J. Edmonds and E. Johnson, "Matching, Eulerian Tours, and the Chinese Postman," *Mathematical Programming*, vol 5 (1973), pp. 88-124.

[11] B. Jackson, "Shortest Circuit Covers and Postman Tours in Graphs with a Nowhere Zero 4-Flow," *SIAM J. Computing* 19,4 (1990), pp. 659-665.

[12] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman Publishers, 1979.