Handbook in Operations Research and Management Science

Volume on “Networks”
Chapter on “Design of Survivable Networks”

by
M. Grötschel*
C.L. Monma**
M. Stoer*

March 1993

*Konrad-Zuse-Zentrum für
Informationstechnik Berlin
Heilbronner Str. 10
D-1000 Berlin 31, Germany

**Bellcore
445 South Street
Morristown, New Jersey 07960

Contents

1 Overview

This chapter focuses on the important practical and theoretical problem of designing survivable communication networks, i.e., communication networks that are still functional after the failure of certain network components. We motivate this topic in Section 2 by using the example of fiber optic communication network design for telephone companies. A very general model (for undirected networks) is presented in Section 3 which includes practical, as well as theoretical, problems, including the well-studied minimum spanning tree, Steiner tree, and minimum cost k-connected network design problems.

The development of this area starts with outlining structural properties in Section 4 which are useful for the design and analysis of algorithms for designing survivable networks. These lead to worst-case upper and lower bounds. Heuristics that work well in practice are also described in Section 4. Polynomially-solvable special cases of the general survivable network design problem are summarized in Section 5.

Section 6 contains polyhedral results from the study of these problems as integer programming models. We present large classes of valid inequalities and indicate when they are facet-defining. We also summarize the complexity of the separation problem for these inequalities. Finally we provide complete and nonredundant descriptions of a number of polytopes related to network survivability problems of small dimensions.

Section 7 contains computational results using cutting plane approaches based on the polyhedral results of Section 6 and the heuristics described in Section 4. The results show that these methods are efficient and effective in producing optimal or near-optimal solutions to real-world problems.

A brief review of the work on survivability models of directed networks is given in Section 8. We also show here how directed models can help to solve undirected cases.

2 Motivation

In this section we set the stage for the topic of this chapter by considering an application to designing communication networks for telephone companies based on fiber optic technology. We will use this to introduce the concept of survivability in network design and to motivate the general optimization models described in the next section. It will become clear later that our models capture many other situations that arise in practice and in theory as well.

Fiber optic technology is rapidly being deployed in communication networks throughout the world because of its nearly-unlimited capacity, its reliability and cost-effectiveness. The high capacity of new technology fiber transmission systems has resulted in the capability to carry many thousands of telephone conversations and high-speed data on a few strands of fiber. These advantages offer the prospect of ushering in many new information networking services which were previously either technically impossible or economically infeasible.

The economics of fiber systems differ significantly from the economics of traditional copper-based technologies. Copper-based technologies tend to be bandwidth limited. This results in a mesh-like network topology, which necessarily has many diverse paths between any two locations with each link carrying only a very small amount of traffic. In contrast, the high-capacity fiber technologies tend to suggest the design of sparse "tree-like" network topologies. These topologies have only a few diverse paths between locations (often just a single path) and each link has a very high traffic volume. This raises the possibility of significant service disruptions due to the failure of a single link or single node in the network. The special report "Keeping the phone lines open" by Zorpette (1989) describes the many man-made and natural causes that can disrupt communication networks, including fires, tornados, floods, earthquakes, construction activity and terrorist activities. Such failures occur surprisingly frequently and with devastating results as described in this report and in the popular press (e.g., see Newark Star Ledger (1987,1988a,1988b), New York Times (1988,1989), Wall Street Journal (1988)).

Hence, it is vital to take into account such failure scenarios and their potential negative consequence when designing fiber communication networks. Recall that one of the major functions of a communication network is to provide connectivity between users in order to provide a desired service. We use the term "survivability" to mean the ability to restore network service in the event of a catastrophic failure, such as the complete loss of a transmission link or a facility switching node. Service could be restored by means of routing traffic around the damage through other existing facilities and switches, if this contingency is provided for in the network architecture. This requires additional connectivity in the network topology and a means to automatically reroute traffic after the detection of a failure.

A network topology could provide protection against a single link failure if it remains connected after the failure of any single link. Such a network is called "two-edge connected" since at least two edges have to be removed in order to disconnect the network. However, if there is a node in the network whose removal does disconnect the network, such a network would not protect against a single node failure. Protection against a single node failure can be provided in an analogous manner by "two-node connected" networks.

In the case of fiber communication networks for telephone companies, two-connected topologies provide an adequate level of survivability since most failures usually can be repaired relatively quickly and, as statistical studies have revealed, it is unlikely that a second failure will occur in the duration. However, for other applications it may be necessary to provide higher levels of connectivity.

One simple and cost effective means of using a two-connected topology to achieve an automatic recovery from a single failure is called diverse protection routing. Most fiber transmission systems employ a protection system to back up the working fiber systems. An automatic protection switch detects failure of a working system and switches the working service automatically to the protection system. By routing the protection system on a physically diverse route from the working system, one provides automatic recovery from a single failure at the small cost of a slightly longer path for the protection system than if it were routed on the same physical path as the working system. One would suspect, and studies have proven, that the additional cost of materials and installation of diverse fiber routes would be acceptable for the benefits gained (see Wu, Kolar and Cardwell (1988a, 1988b), Kolar and Wu (1988), and Cardwell, Wu and Woodall (1988)).

We consider the problem of designing minimum cost networks satisfying certain connectivity requirements that model network survivability. A formal model will be described in the next section. This model allows for different levels of survivability that arise in practical and theoretical models including the minimum spanning tree, Steiner tree, and minimum cost k-connected network design problems.

The application described here requires three distinct types of nodes: special offices, which must be protected from single edge or node failures, ordinary offices, which need only be connected by a single path, and optional offices, which may be included or excluded from the network design depending only upon cost considerations. The designation of office type is performed by a planner based on a cost/benefit analysis. Normally, the special offices are highly important and/or high-revenue-producing offices, with perhaps a high proportion of priority services. It may not be economically possible to ensure the service of ordinary or optional offices in the face of potential failures. In fact, some offices only have one outlet to the network and so it would be technologically impossible to provide recovery if this path were blocked by a network failure.

We note that two-connected network topologies are cost effective. For example, a typical real-world problem which we solved has a survivable network with cost of only 6% above the minimum spanning tree; however, a single failure in the tree network could result in the loss of 33% of the total traffic while the survivable network could lose at most 3% from any single failure. We also note that the heuristic methods described in Section 4 and the polyhedral cutting plane methods described in Section 6 are efficient and very effective in generating optimal and near-optimal network topologies as we describe in Section 7.

We conclude this section by pointing out that the topology design problem considered here is just the first step in the overall design of fiber communications networks. There are other issues that need to be addressed once the topology is in place. For instance, demands for services are usually in units of circuits, called DS0 rate, or bundles of 24 circuits, called DS1 rate. Fiber optic transmission rates come in units of 28 DS1s or 672 DS0s, called DS3 rate. Hence, it is necessary to place multiplexing equipment to convert between the three rates, and to route these demands through the network. Another issue is that the fiber signals need to be amplified using repeater equipment if the signals travel beyond a given threshold distance.

Furthermore, the network is generally organized into a facility hierarchy. That is, offices are grouped together into clusters, with each cluster having one hub office to handle traffic between clusters. This grouping considers such factors as community-of-interest and geographic area. Groups of clusters can be grouped into sectors, with each sector having one gateway office, which is a hub building designated to handle inter-sector traffic. This hierarchy allows traffic to be concentrated into high capacity routes to central locations where the demands are sorted according to destination. This concept of hub routing has proven to give near-optimal results, see Wu and Cardwell (1988). A two-connected network allows for dual homing, i.e., for the possibility of splitting the demand at an office between a home hub and foreign hub to protect against a hub failure.

Given the complexity of the overall design problem, these issues are generally handled in a sequential fashion with the topology question being decided first. This process is often iterated until an acceptable network design is obtained. In this chapter, we only deal with the network topology design aspect and not the multiplexing or bundling aspects. For an overview of this combined process and a description of a computer-based planning tool (Bellcore (1988)) incorporating all of these features, see Cardwell, Monma and Wu (1989).

3 Integer Programming Models of Survivability

In this section, we formalize the undirected survivable network design problems that are being considered in this chapter. Variants that are based on directed networks will be briefly treated in Section 8.

3.1 The General Model for Undirected Networks

A set V of **nodes** is given representing the locations of the offices that must be interconnected into a network in order to provide the desired services. A collection E of **edges** is also specified that represent the possible pairs of nodes between which a direct transmission link can be placed. We let $G = (V, E)$ be the (undirected) graph of possible direct link connections. Each edge $e \in E$ has a nonnegative **fixed cost** c_e of establishing the direct link connection. For our range of applications, loops are irrelevant. Thus we assume in the following that graphs have no loops. Parallel transmission links occur in practice; therefore, our graphs may have parallel edges. For technical reasons to be explained later, we will however restrict ourselves in this paper to simple graphs (i.e., loopless graphs without parallel edges) when we consider node survivability problems and node connectivity.

The cost of establishing a network consisting of a subset $F \subseteq E$ of edges is the sum of the costs of the individual links contained in F. The goal is to build a minimum-cost network so that the required survivability conditions, which we describe below, are satisfied. We note that the cost here represents setting up the topology for the communication network and includes placing conduits in which to lay the fiber cable, placing the cables into service, and other related costs. We do not consider costs that depend on how the network is implemented such as routing, multiplexing, and repeater costs. Although these costs are also important, it is (as mentioned in Section 2) usually the case that a topology is designed first and then these other costs are considered in a second stage of optimization.

If $G = (V, E)$ is a graph, $W \subseteq V$ and $F \subseteq E$, then we denote by $G - W$ and $G - F$ the graph that is obtained from G by deleting the node set W and the edge set F, respectively. For notational convenience we write $G - v$ and $G - e$ instead of $G - \{v\}$ and $G - \{e\}$, respectively. The difference of two sets M and N is denoted by $M \setminus N$.

For any pair of distinct nodes $s, t \in V$, an $[s,t]$-**path** P is a sequence of nodes and edges $(v_0, e_1, v_1, e_2, \ldots, v_{l-1}, e_l, v_l)$, where each edge e_i is incident with the nodes v_{i-1} and $v_i (i = 1, \ldots, l)$, where $v_0 = s$ and $v_l = t$, and where no node or edge appears more than once in P. A collection $P_1, P_2, \ldots, P_k$ of $[s,t]$-paths is called **edge-disjoint** if no edge appears in more than one path, and is called **node-disjoint** if no node (except for s and t) appears in more than one path. In standard graph theory two parallel edges are not considered as node disjoint paths. For our applications it is sensible to do so. However, this modification would lead to considering nonstandard variations of node connectivity, reformulations of Menger's theorem etc. In order not to trouble the reader with these technicalities we have decided to restrict ourselves to simple graphs when node disjoint paths are treated. The results presented in this paper carry, appropriately stated, over to the case where parallel edges are considered as node disjoint paths. This theory is developed in Stoer (1992).

Two different nodes s, t of a graph are called k-**edge** (resp. k-**node**) **connected** if there are k edge-disjoint (resp. node-disjoint) paths between s and t. A graph with at least two nodes is called k-edge or k-node connected if all pairs of distinct nodes of G are k-edge or k-node connected, respectively. For our purposes, the graph K_1 consisting of just one node is k-edge and k-node connected for every natural number k. For a graph $G \neq K_1$, the largest integer k such that G is k-edge connected (resp. k-node connected) is denoted by $\lambda(G)$ (resp. $\kappa(G)$) and is called the **edge connectivity** (resp. **node connectivity**) of G.

The survivability conditions require that the network satisfy certain edge and node connectivity requirements. To specify these, three nonegative integers r_{st}, k_{st} and d_{st} are given for each pair of distinct nodes $s, t \in V$. The numbers r_{st} represent the **edge survivability requirements**, and the numbers k_{st} and d_{st} represent the **node survivability requirements**; this means that the network $N = (V, F)$ to be designed has to have the property that, for each pair $s, t \in V$ of distinct nodes, N must contain at least r_{st} edge disjoint $[s, t]$-paths, and that the removal of at most k_{st} nodes (different from s and t) from N must leave at least d_{st} edge disjoint $[s, t]$-paths. (Clearly, we may assume that $k_{st} \leq |V| - 2$ for all $s, t \in V$, and we will do this throughout this chapter). These conditions ensure that some communication path between s and t will survive a prespecified level of combined failures of both nodes and links. The levels of survivability specified depend on the relative importance placed on maintaining connectivity between different pairs of offices.

Let us now introduce a variable x_e for each edge $e \in E$, and consider the vector space R^E. Every subset $F \subseteq E$ induces an **incidence vector** $\chi^F = (\chi_e^F)_{e \in E} \in R^E$ by setting $\chi_e^F := 1$ if $e \in F$, $\chi_e^F := 0$ otherwise; and vice versa, each 0/1-vector $x \in R^E$ induces a subset $F^x := \{e \in E \mid x_e = 1\}$ of the edge set E of G. If we speak of the incidence vector of a path in the sequel we mean the incidence vector of the edges of the path. We can now formulate the network design problem introduced above as the following integer linear program.

(3.1) $\quad \min \sum_{ij \in E} c_{ij} x_{ij}$

subject to

(i)	$\sum_{i \in W} \sum_{j \in V \setminus W} x_{ij} \geq r_{st}$	for all pairs $s, t \in V, s \neq t$ and for all $W \subseteq V$ with $s \in W, t \notin W$;
(ii)	$\sum_{i \in W} \sum_{j \in V \setminus (Z \cup W)} x_{ij} \geq d_{st}$	for all pairs $s, t \in V, s \neq t$; and for all $Z \subseteq V \setminus \{s, t\}$ with $\lvert Z \rvert = k_{st}$ and for all $W \subseteq V \setminus Z$; with $s \in W, t \notin W$;
(iii)	$0 \leq x_{ij} \leq 1$	for all $ij \in E$;

(iv) $\quad x_{ij}$ integral $\qquad\qquad\qquad$ for all $ij \in E$.

Note that if, for each pair s, t of distinct nodes in V and for each set $Z \subseteq V \setminus \{s,t\}$ with $|Z| = k_{st}$, $N - Z$ contains at least d_{st} edge disjoint $[s,t]$-paths, then all node survivability requirements are satisfied, i.e., inequalities of type (ii) need not be considered for node sets $Z \subseteq V \setminus \{s,t\}$ with $|Z| < k_{st}$. It follows from Menger's theorem (see Chapter ??) that, for every feasible solution x of (3.1), the subgraph $N = (V, F^x)$ of G defines a network that satisfies the given edge and node survivability requirements.

This model, introduced in Grötschel & Monma (1990), generalizes and unifies a number of problems that have been investigated in the literature either from a practical or theoretical point of view. We mention here some of these cases.

The classical *network synthesis problem* for multiterminal flows (see Chapter 13) is obtained from (3.1) by dropping the constraints (ii) and (iv). In the standard formulation of the network synthesis problem, the upper bounds $x_e \leq 1$ are not present. But our model allows parallel edges in the underlying direct-link graph, or equivalently, allows the upper bound in constraints (iii) to take on any nonnegative values for each edge. This linear programming problem has a number of constraints that is exponential in the number of nodes of G. For the case $c_{ij} = c$ for all $ij \in E$, where c is a constant, Gomory & Hu (1961) found a simple algorithm for its solution. Bland, Goldfarb & Todd (1981) pointed out that the separation problem for the class of inequalities (i) can be solved in polynomial time by computing a minimum capacity cut; thus, it follows by the ellipsoid method that the classical network synthesis problem can be solved in polynomial time. (See Grötschel, Lovász and Schrijver (1988) for details on the ellipsoid algorithm and its applications to combinatorial optimization.) The *integer* network synthesis problem, i.e., the problem obtained from (3.1) by dropping constraints (ii) and the upper bounds $x_e \leq 1$, was solved by Chou & H. Frank (1970), for the case $c_{ij} = c$ for all $ij \in V \times V$, and $r_{ij} \geq 2$ for all ij.

The *minimum spanning tree problem* can be phrased as the task to find a minimum-cost connected subset $F \subseteq E$ of edges spanning V (see Chapter 12). This problem can be viewed as a special case of (3.1) as follows. We drop the constraints (ii) and set $r_{st} := 1$ for all distinct $s, t \in V$ in constraints (i).

Similarly, the closely-related *Steiner tree problem* is to find a minimum-cost connected subset $F \subseteq E$ of edges which span a specified subset $S \subseteq V$ of nodes. This problem is a special case of (3.1) where we drop constraints (ii) and set $r_{st} := 1$ in constraints (i), for all $s, t \in S$, and $r_{st} := 0$ otherwise. (See also Chapter 12.) Let us remark at this point that the Steiner tree problem is well-known to belong to the class of NP-hard problems. As it is a special case of (3.1), our general problem of designing survivable networks is NP-hard as well. (See Garey & Johnson (1979) for details on the theory of NP-completeness.)

The problem of finding *a minimum-cost k-edge connected network* in a given graph is a special case of (3.1) where all inequalities (ii) are dropped and where $r_{st} = k$ for all distinct $s, t \in V$. The problem of finding an *optimal k-node connected network*, for $k \leq |V| - 1$, is a special case of (3.1) where we drop the constraints (i) and set $k_{st} := k - 1$ and $d_{st} := 1$ for all distinct $s, t \in V$.

3.2 A Brief Discussion of the Model

The rest of this chapter is mainly devoted to studying various aspects of model (3.1) and some of its special cases. Among other subjects, we describe heuristics to compute upper and lower bounds for the optimum value of (3.1). To compute a lower bound one is naturally led to dropping the integrality constraints (iv) and solving the LP-relaxation (3.1) (i), (ii), (iii) of the survivable network design problem.

Two questions immediately arise. Can one solve this LP despite the fact that it has exponentially many constraints? Can one find a better (or a series of increasingly better) LP-relaxations that are solvable in polynomial (or practically reasonable) time?

We will address the first question in Section 7 and the second in Section 6. But here we would like to give a glimpse at the approach that leads to answering these questions. The method involved is known as polyhedral combinatorics. It is a vehicle to provide (in some sense) the best possible LP-relaxation. We want to demonstrate now how the second question leads to the investigation of certain integral polyhedra in a very natural way. See Grötschel & Padberg (1985) and Pulleyblank (1990) for a survey on polyhedral combinatorics.

To obtain a better LP-relaxation of (3.1) than the one arising from dropping the integrality constraints (iv), we define the following polytope. Let $G = (V, E)$ be a graph, let $E_V := \{st \mid s, t \in V, s \neq t\}$, and let $r, k, d \in Z_+^{E_V}$ be given. Then

$$\text{(3.2)} \qquad \mathrm{CON}(G; r, k, d) := \mathrm{conv}\{x \in \mathbf{R}^E \mid x \text{ satisfies (i)}, \ldots, \text{(iv) of (3.1)}\}$$

is the polytope associated with the network design problem given by the graph G and the edge and node survivability requirements r, k, and d. (Above "conv" denotes the convex hull operator.) In the sequel, we will study $\mathrm{CON}(G; r, k, d)$ for various special choices of r, k and d. Let us mention here a few general properties of $\mathrm{CON}(G; r, k, d)$ that are easy to derive.

Let $G = (V, E)$ be a graph and $r, k, d \in \mathbf{Z}_+^{E_V}$ be given as above. We say that $e \in E$ is **essential with respect to** $(G; r, k, d)$ (short: $(G; r, k, d)$-**essential**) if $\mathrm{CON}(G - e; r, k, d) = \emptyset$. In other words, e is essential with respect to $(G; r, k, d)$ if its deletion from G results in a graph such that at least one of the survivability requirements cannot be satisfied. We denote the set of edges in E that are essential with respect to $(G; r, k, d)$ by $\mathrm{ES}(G; r, k, d)$. Clearly, for all subsets $F \subseteq E \setminus \mathrm{ES}(G; r, k, d)$, $\mathrm{ES}(G; r, k, d) \subseteq \mathrm{ES}(G - F; r, k, d)$ holds. Let $\dim(S)$ denote the **dimension** of a set $S \subseteq \mathbf{R}^n$, i.e., the maximum number of affinely independent elements in S minus 1. Then one can easily prove the following two results (see Grötschel and Monma (1990)).

(3.3) Theorem. *Let $G = (V, E)$ be a graph and $r, k, d \in Z_+^{E_V}$ such that* $\mathrm{CON}(G; r, k, d) \neq \emptyset$. Then

$$\mathrm{CON}(G; r, k, d) \subseteq \{x \in \mathbf{R}^E \mid x_e = 1 \text{ for all } e \in \mathrm{ES}(G; r, k, d)\},$$

$$\dim(\mathrm{CON}(G; r, k, d)) = |E| - |\mathrm{ES}(G; r, k, d)|. \square$$

An inequality $a^T x \leq \alpha$ is **valid** with respect to a polyhedron P, if $P \subseteq \{x \mid a^T x \leq \alpha\}$; the set $F_a := \{x \in P \mid a^T x = \alpha\}$ is called the **face** of P defined by $a^T x \leq \alpha$. If $\dim(F_a) = \dim(P) - 1$ and $F_a \neq \emptyset$ then F_a is a **facet** of P, and $a^T x \leq \alpha$ is called **facet-defining** or **facet-inducing**.

(3.4) Theorem. *Let $G = (V, E)$ be a graph and $r, k, d \in \mathbf{Z}_+^{E_V}$ such that* $\mathrm{CON}(G; r, k, d) \neq \emptyset$. *Then*

(a) *$x_e \leq 1$ defines a facet of* $\mathrm{CON}(G; r, k, d)$ *if and only if $e \in E \setminus$* $\mathrm{ES}(G; r, k, d)$;

(b) *$x_e \geq 0$ defines a facet of $CON(G; r, k, d)$ if and only if $e \in E \setminus$* $\mathrm{ES}(G; r, k, d)$ *and* $\mathrm{ES}(G; r, k, d) = \mathrm{ES}(G - e; r, k, d)$. $\square$

Theorems (3.3) and (3.4) solve the dimension problem and characterize the trivial facets. But these characterizations are (in a certain sense) algorithmically intractible as the next observation shows which follows from results of Ling and Kameda (1987).

(3.5) Remark. *The following three problems are* NP-*hard.*

Instance: A graph $G = (V, E)$ and vectors $r, k, d \in \mathbf{Z}_+^E$.

Question 1: Is $\mathrm{CON}(G; r, k, d)$ *empty?*

Question 2: Is $e \in E$ $(G; r, k, d)$-essential?

Question 3: What is the dimension of $\mathrm{CON}(G; r, k, d)$? $\square$

However, for most cases of practical interest in the design of survivable networks, the sets $\mathrm{ES}(G; r, k, d)$ of essential edges can be determined easily, and thus the trivial LP-relaxation of (3.1) following from (3.3) and (3.4) can be set up without difficulties. (We will comment on this in the sequel.)

Before continuing, let us remark that there is an easy way to improve upon the formulation of (3.1) by excluding a number of inequalities that are obviously redundant.

Given $G = (V, E)$ and $r, k, d \in \mathbf{Z}_+^{E_V}$, extend the functions r and d to functions operating on sets by setting

$$\text{(3.6)} \qquad \mathrm{con}(W) := \max\{r_{st} \mid s \in W, t \in V \backslash W\}$$

and

$$\text{(3.7)} \qquad d(Z, W) := \max\{d_{st} \mid s \in W \backslash Z, t \in V \backslash (Z \cup W), k_{st} \geq |Z|\} \ \text{ for } Z, W \subseteq V.$$

We call a pair (Z, W) with $Z, W \subseteq V$ **eligible** (with respect to k) if $Z \cap W = \emptyset$ and $|Z| = k_{st}$ for at least one pair of nodes with $s \in W$ and $t \in V \setminus (Z \cup W)$. Then $\mathrm{CON}(G; r, k, d)$ is clearly contained in the solution set of the following system of equations and inequalities.

(3.8)	(i)	$\sum_{i \in W} \sum_{j \in V \backslash W} x_{ij} \geq \mathrm{con}(W)$	for all $W \subseteq V, \emptyset \neq W \neq V$;
	(ii)	$\sum_{i \in W} \sum_{j \in V \backslash (Z \cup W)} x_{ij} \geq d(Z, W)$	for all eligible pairs (Z, W) of subsets of V;
	(iii)	$x_{ij} \leq 1$	for all $i_j \in E \backslash ES(G; r, k, d)$;
	(iv)	$x_{ij} = 1$	for all $ij \in \mathrm{ES}\ (G; r, k, d)$;
	(v)	$x_{ij} \geq 0$	for all $E \setminus\ \mathrm{ES}(G; r, k, d)$ with $\mathrm{ES}(G; r, k, d) = \mathrm{ES}(G - e; r, k, d)$.

Of course, $\mathrm{CON}(G; r, k, d)$ is the convex hull of the integral solution of (3.8).

3.3 A Model Used in Practice

The model discussed so far mixes node and edge connectivity, and provides, for each pair of nodes, the possibility to specify particular connectivity requirements. It is, as mentioned before, a quite general framework that models many practical situations simultaneously. This generality demands a considerable amount of data. In our real-world application, it turned out that the network designers were either interested in node connectivity or in edge connectivity requirements but not in both simultaneously. Also, the data for implementing the general model were not available in practice. A specialized version, to be described below, proved to be acceptable from the point of view of data acquisition and was still considered a reasonable model of reality by practitioners.

To model these (slightly more restrictive) survivability conditions, we introduce the concept of node types. For each node $s \in V$ a nonnegative integer r_s, called the **type** of s, is specified. For any $W \subseteq V$, the integer $r(W) := \max\{r_v \mid V \in W\}$ is called the **type** of W. We say that the network $N = (V, F)$ to be designed satisfies the **node survivability conditions** if, for each pair $s, t \in V$ of distinct nodes, N contains at least

$$r_{st} := \min\{r_s, r_t\} \tag{3.9}$$

node disjoint $[s, t]$-paths. Similarly, we say that $N = (V, F)$ satisfies the **edge survivability conditions** if, for each pair $s, t \in V$ of distinct nodes, N contains r_{st} edge disjoint $[s, t]$-paths. These conditions ensure that some communication path between s and t will survive a prespecified level of node or link failures. We will discuss these special cases in more detail later. To make the (somewhat clumsy) general notation easier, we introduce further symbols and conventions to denote these node- or edge-survivability models. Given a graph $G = (V, E)$ and a vector of node types $r = (r_s)_{s \in V}$ we assume – without loss of generality – that there are at least two nodes of the largest type. If we say that we consider the kNCON problem (for G and r) then we mean that we are looking for a minimum-cost network that satisfies the node survivability conditions and where $k = \max\{r_s | s \in V\}$. Similarly, we speak of the kECON problem (for G and r), when we consider the edge survivability conditions.

Let $G = (V, E)$ be a graph. For $Z \subseteq V$, let $\delta_G(Z)$ denote the set of edges with one endnode in Z and the other in $V \setminus Z$. It is customary to call $\delta_G(Z)$ a **cut**. If it is clear with respect to which graph a cut $\delta_G(Z)$ is considered, we simply drop the index and write $\delta(Z)$. We also write $\delta(v)$ for $\delta(\{v\})$. If X, Y are subsets of V with $X \cap Y = \emptyset$, we set $[X : Y] := \{ij \in E \mid i \in X, j \in Y\}$, thus $\delta(X) = [X : V \setminus X]$. For any subset of edges $F \subseteq E$, we let $x(F)$ stand for the sum $\sum_{e \in F} x_e$. Consider the following integer linear program for a graph $G = (V, E)$ with edge costs c_e for all $e \in E$ and node types r_s for all $s \in V$ (using (3.9) in the definition of $\mathrm{con}(W)$ in (3.6)):

(3.10) $\min c^T x$

subject to

(i)	$x(\delta(W)) \geq \text{con}(W)$	for all $W \subseteq V$, $\emptyset \neq W \neq V$;
(ii)	$x(\delta_{G-Z}(W)) \geq \text{con}(W) - \lvert Z\rvert$	for all pairs $s, t \in V, s \neq t$, and for all $\emptyset \neq Z \subseteq V \setminus \{s, t\}$ with $\lvert Z\rvert \leq r_{st} - 1$, and for all $W \subseteq V \setminus Z$ with $s \in W, t \notin W$;
(iii)	$0 \leq x_{ij} \leq 1$	for all $ij \in E$;
(iv)	x_{ij} integral	for all $ij \in E$.

It follows from Menger's theorem that the feasible solutions of (3.10) are the incidence vectors of edge sets F such that $N = (V, F)$ satisfies all node survivability conditions; i.e., (3.10) is an integer programming formulation of the kNCON problem. Deleting inequalities (ii) from (3.10) we obtain, again from Menger's theorem, an integer programming formulation for the kECON problem. The inequalities of type (i) will be called **cut inequalities** and those of type (ii) will be called **node cut inequalities.**

The polyhedral approach to the solution of the kNCON (and similarly the kECON) problem consists of studying the polyhedron obtained by taking the convex hull of the feasible solutions of (3.9). We set

$$\text{NCON}(G; r) := \text{conv}\{x \in \mathbf{R}^E | x \text{ satisfies } (3.10)(\text{i}), \ldots, (\text{iv})\},$$

$$\text{ECON}(G; r) := \text{conv}\{x \in \mathbf{R}^E | x \text{ satisfies. } (3.10)(\text{i}), (\text{iii}), (\text{iv})\}.$$

It will sometimes be convenient to denote these polyhedra by $k\text{NCON}(G; r)$ and $k\text{ECON}$ $(G; r)$, where $k = \max\{r_s | s \in V\}$, since this implicitly provides a notation for the maximum node type. To tie this notation with the previously introduced more general concept, note that

$$k\text{ECON}(G; r) = \text{CON}(G; r', 0, 0),$$

where $r' \in \mathbf{R}^{V \times V}$ with $r'_{st} := \min\{r_s, r_t\}$ for all $s, t \in V$. Also, if there are no parallel edges

$$k\text{NCON}(G; r) = \text{CON}(G; r', k', d'),$$

where $k'_{st} := \max\{0, r'_{st} - 1\}$ for all $s, t \in V$ and $d' := r' - k'$.

This survivability model, the polyhedra, and the integer and linear programming problems associated with the kECON and kNCON problems, will be studied in more detail in the sequel.

4 Structural Properties and Heuristics

In this section we describe some heuristic approaches for the solution of ECON and NCON problems. There are standard methods like greedy and interchange heuristics and heuristics that are motivated by techniques for the approximate solution of other NP-hard problems, like the (much investigated) traveling salesman problem. Some heuristics are more special-purpose since they make use of structural properties of k-connected graphs. A few structural results and their uses are reviewed in Sections 4.1 and 4.2. Section 4.2 concentrates on the design of practically-effective heuristics for ECON and NCON problems, while Section 4.3 discusses lower bounds and heuristics with worst-case performance guarantees.

4.1 Lifting and the Structure of Optimum Solutions

Connectivity is a very rich and active topic of graph theory and it is conceivable that the knowledge that has been accumulated in this area can be exploited further for the design of effective approximation algorithms. We do not attempt to cover the work on connectivity here in detail and refer to A. Frank (1992) for a comprehensive survey of connectivity results. We will just mention a few structural properties of connected graphs that have been employed for the design of practically-useful heuristics. We begin by describing a "local" construction technique. Lovász (1976) and Mader (1978) have proved so-called "lifting theorems" that show that simple manipulations of a graph can be made without destroying certain edge connectivity properties. These manipulations are useful for construction heuristics for the ECON problem (where parallel edges are allowed) for general connectivity requirements r_{vw}.

In order to state these results we have to introduce a few definitions. Let $G = (V, E)$ be a graph, and let x be a node of G that we call *special*. We assume that x is adjacent to distinct nodes y and z. The graph G' obtained from G by deleting the edges xy and xz and adding the edge yz is called *lifting* of G at x.

If the edge-connectivity of G' between any two nodes of $V' - x$ is not smaller than that of G, then the lifting is called **admissible**.

(4.1) Theorem. *[Mader 1978]. Let $G = (V, E)$ be a graph with special node x.*

(a) *If the degree of x is at least 4 and x is not a cut node then G has an admissible lifting at x.*

(b) *If x is a cut node and no single edge incident to x is a cut then G has an admissible lifting at x.* □

(4.2) Theorem. *[Lovász 1976]. Let $G = (V, E)$ be an Eulerian graph, let x be a (special) node of even degree, and let y be any node adjacent to x in G. Then there is another neighbor node z of x, so that the lifting at x involving y and z is admissible.* □

Consider the kECON and kNCON problems for the complete graph K_n where $r_x = k$ for all nodes x, and where the nonnegative costs satisfy the triangle inequality, i.e., $c_{xz} \leq c_{xy} + c_{yz}$ for all nodes x, y and z. Using the Lifting Theorems (4.1) and (4.2), Monma, Munson and Pulleyblank (1990) showed the following result for kECON for K_n with $k = 2$.

(4.3) Theorem. *Given costs satisfying the triangle inequality, there is an optimal k-edge connected spanning network satisfying the following conditions:*

(a) *all nodes are of degree k or $k + 1$; and*

(b) *removing any set of at most k edges does not leave all of the resulting connected components k-edge connected.* □

It is easy to see that if the cost function satisfies the triangle inequalities, there is an optimal 2-node connected solution with cost equal to an optimal 2-edge connected solution. In fact, Monma, Munson and Pulleyblank (1990) show that the term "k-edge" can be replaced by "k-node" throughout (4.3) to obtain a similar result for kNCON for K_n with $k = 2$. Furthermore, they show that these "characterize" the optimal 2-connected networks in the following sense: given any 2-connected graph $G = (V, E)$ satisfying conditions (4.3) (a) and (b) for $k = 2$, then there exist costs satisfying the triangle inequality such that G is the unique optimal solution.

Bienstock, Brickell and Monma (1990) showed that (4.3) holds for arbitrary k. They also showed that "k-edge" can be replaced by "k-node" throughout (4.3) with the technical restriction that $|V| \geq 2k$ in order for (a) to hold. This technical restriction is necessary since without it there is an infinite family of examples where condition (a) fails.

We note that the cost of an optimal k-edge connected solution may be strictly less than the cost of an optimal k-node connected solution for any $k \geq 3$, and that the conditions in (4.3) do not characterize the optimal solutions as they do in the case $k = 2$. The proof of (4.3) for the k-edge connected case uses the Lifting Theorem (4.1). The proof of (4.3) in the k-node connected case is much more difficult and requires the use of pairs of liftings as well as a number of further technical results.

4.2 Construction and Improvement Heuristics

We now describe some heuristics for constructing feasible networks and heuristics for improving the cost of a feasible solution for the kECON and kNCON problems. This is, to a large extent, a summary of the work of Monma and Shallcross (1989) for the "low-connected" survivable network design problem, that is, problems with node types in $\{0, 1, 2\}$. The performance on real-world problems is described in Section 7. These heuristics were inspired by the wide variety of heuristics for the traveling salesman problem and other combinatorial optimization problems and were designed to work on sparse underlying graphs. The heuristics are used in a local search approach to obtain low-cost network designs; see Papadimitriou and Steiglitz (1982) for a general discussion of local search procedures for combinatorial optimization problems. It is obvious how these heuristics can be generalized and applied to the general survivable network design problem, and so we just briefly mention one such extension.

One useful structural fact is that any edge-minimal two-connected graph $G = (V, E)$ can be constructed by an "ear decomposition" procedure; see Lovász and Plummer (1986). That is, first find a cycle C in G. Then repeatedly find a path P, called an ear, that starts at a node v in the solution, passes through nodes not yet in the solution, and ends up at a node w in the solution. All edge-minimal two-edge connected graphs can be constructed in this manner. If the nodes v and w are required to be distinct, then every edge-minimal two-node connected graph can be constructed in this manner.

The ear decomposition approach can be employed to construct a feasible 2-connected subgraph of a graph $G = (V, E)$ using costs to add ears in a greedy fashion. We call this the **greedy ears construction heuristic.** The first step is to construct a partial solution consisting of a cycle C spanning the set of nodes of type 2, called the "special nodes". This is done by randomly selecting a special node v, and then selecting a special node w whose shortest path P to v is the longest among all special nodes. Let node u be the node next to w on the path P. We will construct a short cycle through the edge uw by finding a shortest path from u to w not using the edge uw. (There must be such a path; if not, there would not two node-disjoint paths between the special nodes v and w and so the problem would be infeasible). The next step is to repeatedly add "short" ears to the current partial solution until all special nodes are on this two-connected network. This is done by first selecting a special node z, not yet in the solution, whose shortest path P to the partial solution is longest among all special nodes not yet included. We will find another shortest path Q from z to the partial solution that does not use any edges of P and that terminates on the partial solution at a node w other than v. (Again, such a path must exist for the problem to be feasible). The combination of paths P and Q must contain an ear which is added to the partial solution.

A second construction heuristic uses the ear decomposition approach to construct a feasible 2-connected subgraph of a graph $G = (V, E)$ in a random fashion. We call this the **random sparse construction heuristic**. The first step is to construct an initial random cycle C spanning a subset of the special nodes. This is done by randomly choosing a special node v, and constructing a depth-first-search tree T rooted at v. Form a cycle by randomly choosing an edge of the form vz that is not in T. (There must be such an edge or else v is not on any cycle and so the problem is infeasible.) Next, random ears are repeatedly added until all special nodes are on the two-connected network. This is done by first constructing a depth-first-search forest F rooted at the nodes that are in the partial solution. A node v is said to be *allowed* if v is not yet in the solution, but it has an edge vw in E but not F, where w is in the solution. Randomly choose a node v from among the allowed nodes. (Again, there must be such a node or the problem is infeasible.) Let T be the tree in the forest F containing v, and let z be the root of T. The random ear chosen is the path from v to z in T, together with the edge vw. Since this method does not use cost information, generally it does not produce a low cost solution. However, this method is useful for generating starting random initial solutions on which to apply the improvement methods.

Improvement heuristics apply local transformations to any feasible network in order to reduce the cost while preserving feasibility. These transformations are applied until a locally optimal network is obtained; that is, no further improvements of these types are possible. Six local transformations are described in the sequel. These transformations are general enough to cover a wide range of feasible topologies, yet fast enough to be performed in real time, even on a personal computer.

Every two-connected network contains at least one cycle, and often is made up of many interconnected cycles. Furthermore, replacing the edges in a cycle C by edges forming a cycle C' on the same nodes preserves the feasibility of a solution. So it is natural to draw upon the extensive research on the traveling salesman problem (see Lawler et al. (1985)) for finding a near-optimal cycle. This is the basis of the two-optimal cycle and three-optimal cycle improvement heuristics. The pretzel, quetzel and degree improvement heuristics alter the structure of the two-connected part of the solution in less obvious ways. The one-optimal improvement heuristic alters the structure of the entire solution. These heuristics are described below.

The **two-optimal interchange heuristic** attempts to replace two edges of a cycle C by two edges not in the cycle to form a new cycle C' of lower cost. Similarly, the **three-optimal interchange heuristic** attempts to replace three edges of a cycle C by three edges not in the cycle to form a new cycle C' of lower cost.

These improvement heuristics replace one cycle C by another cycle C' on the same nodes, and so do not change the fundamental structure underlying the solution. The **pretzel transformation** replaces an edge uv of a cycle C by two crossing edges ux and vy to form a "pretzel" P, where the nodes u, v, x and y appear in order on the cycle.

The **quetzel transformation** is the reverse of the pretzel transformation; that is, a pretzel P is replaced by a cycle C by removing two crossing edges ux and vy and adding an edge uv.

As mentioned before, if the costs satisfy the triangle inequality, then there is an optimal two-connected solution where all nodes are of degree two or three. The proof of this result employs the Lifting Theorems (4.1) and (4.2). We will describe now the algorithmic use of these theorems in the form of **degree improvement transformations**.

Let node u be of degree four or more, and let nodes a, b, c, and d be four of its neighbors. There are three cases to consider. In Case a, there are node-disjoint paths P_{ab} and P_{bc} from a to b, and from b to c, respectively, which miss node u. In this case, edge ub is a chord and can be deleted. So we may assume that no three of the nodes a, b, c and d have such paths. Therefore, the paths P_{ab} and P_{bc} must intersect in node v and the paths P_{bc} and P_{cd} must intersect in node w. If nodes v and w are different, we are in Case b, and edges bu and uc are removed, and edge bc is added; this preserves the connectivity requirements and does not increase the cost if the triangle inequality holds. If nodes v and w are the same node, we are in Case c. Let a', b', c' and d' be the neighbors of $v(= w)$. Edges $au, ub, b'v$ and vc' are removed, and edges ab and $b'c'$ are added; this preserves the connectivity requirements and does not increase the cost if the triangle inequality holds. In all cases, the degree of node u is decreased, and no other degrees are increased; so repeated application of these transformations guarantees that an optimal solution where all degrees are two or three will be obtained so long as the triangle inequality holds.

All of the previous improvement heuristics operated only on the two-connected part of the solution. The **one-optimal interchange heuristic** considers the entire solution. This heuristic attempts to remove an edge uv from the current feasible solution and replace it with another edge of the from ux not currently in the solution. Such an interchange is made only if the resultant network is feasible and of lower cost.

These heuristics were tested on randomly generated problems and on real-world problems of telephone network design; see Monma and Shallcross (1989) for details. They could decrease the cost of manually constructed solutions by about 10 percent in a test of these methods on a real-world application. Comparison with the optimal solutions computed by the cutting plane algorithm (to be described in Section 7) on the same examples show that the gap was usually very small, about 1 percent of the optimal value. The highest running time for a sparse 116-node problem was 156 seconds on an IBM-PC/AT.

Ko and Monma (1989) modified the low-connectivity heuristics of Monma and Shallcross to the design of k-edge or k-node connected networks. A first feasible solution is constructed either by deleting edges successively from the whole graph while maintaining feasibility, or by adding successively k edge-disjoint $[1, j]$-paths of minimum overall length, for all nodes $j \neq 1$. The output is necessarily k-edge connected. The local transformations for the low-connectivity case carry over to the high-connectivity case, except that the feasibility checks have to be implemented differently.

These heuristics could only be tested on random examples, as real-world examples were not yet available. When the cost of the best heuristic solution was compared with the optimal solution produced by our cutting plane algorithm, the gap was approximately 6 percent. Running times on a VAX 8650 ranged between 13 seconds for dense graphs of 20 nodes and $k = 3$ and 120 seconds for dense graphs of 40 nodes and $k = 5$.

Let us remark that to our knowledge the first heuristics for the design of minimum-cost survivable networks under general connectivity requirements date back to Steiglitz, Weiner and Kleitman (1969). Their heuristic consists of a randomized starting routine and an optimization routine where local transformations are applied to a feasible solution as follows. Given a random ordering of the nodes, the *starting routine* adds edges between the first node with the highest connectivity requirement and the first node with the next highest connectivity requirement. In each step the connectivity requirements are updated. If the solution is feasible, the *optimizing routine* tries to improve this solution by successively replacing one pair of edges with another pair of edges to obtain another feasible solution of lower cost until no more improvements can be made this way.

They applied their heuristics to two real-world problems with 10 nodes and 58 nodes, and connectivity requirements in $\{3, 4, 5, 6\}$ and $\{6\}$, respectively. (Unfortunately, the data are not available any more.) The 58-node problem took about 12 minutes (on a UNIVAC 1108) per local optimum. Since no lower bounds on the optimal value for these problems are given, we cannot say how well these heuristics work.

4.3 Heuristics with Performance Guarantees

The last remark leads us to an important issue: quality guarantees, i.e., worst-case performance and lower bounds. The polyhedral approach, to be described later, can be viewed as a technique to obtain very good lower bounds for the optimum value of a kECON or kNCON problem. But sometimes nice performance guarantees for heuristics or estimates for optimum values can be given with less elaborate techniques.

Let us first relate the 2ECON and 2NCON problems to the traveling salesman problem. Since every Hamiltonian cycle is 2-node (and thus 2-edge) connected, the optimum TSP-value, CTSP say, is not smaller than the optimum value COPT of the 2ECON problem. On the other hand, using (4.3), Monma, Munson and Pulleyblank (1990) were able to show that if the triangle inequality holds, COPT can be bounded by CTSP from below by a constant factor, more precisely

$$\frac{3}{4}\mathrm{CTSP} \leq \mathrm{COPT} \leq \mathrm{CTSP}.$$

To solve the 2ECON problem approximately, Frederickson and JáJá (1982) modified the Christofides heuristic for the traveling salesman problem and proved that, when the triangle inequality holds, the solution attains a cost CCHR with the same worst-case bound as in the traveling salesman problem, namely

$$\mathrm{CCHR} \leq \frac{3}{2}\,\mathrm{COPT}.$$

Another type of lower bound for the k-edge connected network design problem can be derived from the subtour elimination polytope, which is a natural linear programming relaxation of the traveling salesman polytope. Let CSUB denote the value of an optimal solution to the subtour elimination linear program; see Chapter 3. Goemans and Bertsimas (1990) showed that

$$k/2\ \mathrm{CSUB} \leq \mathrm{COPT}.$$

For $k = 2$, this was previously shown by Cunningham, see Monma, Munson and Pulleyblank (1990). These results make use of the Lifting Theorem (4.2).

We conclude this section by describing heuristics by Goemans and Bertsimas (1990) and Agrawal et al. (1991) with worst-case guarantees for the kECON problem. Goemans and Bertsimas developed a tree heuristic with worst-case guarantee for a version of kECON problem with general node types $r \in \mathbf{Z}_+^V$, where edges may be used more than once. This means that a feasible solution to this problem is a vector $x \in \mathbf{Z}_+^E$ of nonnegative integers that satisfies all cut inequalities of $x(\delta(W)) \geq r(W)$, but not necessarily the upper bounds $x_e \leq 1$. A component $x_e \geq 2$ may be interpreted as "edge e used x_e times".

Let $\rho_1 < \rho_2 < \ldots < \rho_p$ be the ordering of the distinct node types in r, let $\rho_0 := 0$, and let V_k be the set of nodes of type at least $\rho_k, k = 1, \ldots, p$.

(4.4) Tree Heuristic

1. *Compute, for all pairs of nodes u, v, the shortest path length c'_{uv} with respect to the given costs c.*
2. *Set $x_e := 0$ for all $e \in V \times V$.*
3. *For $k = 1$ to p do*
   - *Compute $T_k = (V_k, E_k)$ as the minimum spanning tree of the complete graph induced by V_k with respect to costs c'_e.*
   - *Set $x_e := x_e + (\rho_k - \rho_{k-1})$ for all $e \in E_k$.*
4. *Replace each $uv \in V \times V$ with $x_{uv} > 0$ by x_{uv} times the shortest $[u, v]$-path with value c'_{uv}.*

Note that the output of this heuristic is a vector $x \in \mathbf{Z}_+^E$ that satisfies all cut inequalities $x(\delta(W)) \geq r(W)$. Let y be the solution to the LP-relaxation of the ECON problem consisting of cut inequalities (2.4)(i) and nonnegativity constraints $x_e \geq 0$. Goemans and Bertsimas show that

$$\frac{x^T c}{y^T c} \leq \left(2 - \frac{2}{|V_1|}\right) \left(\sum_{k=1}^{p} \frac{\rho_k - \rho_{k-1}}{\rho_k}\right).$$

This bound is tight, if we consider for instance a 2ECON problem on a cycle of costs 1. Goemans and Bertsimas also describe a more refined tree heuristic with a better worst-case guarantee.

Agrawal, Klein, and Ravi (1991) state a heuristic for an ECON problem, where edge-connectivity requirements are given by $r \in \mathbf{Z}_+^{V \times V}$ and the use of multiple parallel edges is allowed in the solution. They prove that their algorithm outputs a solution whose cost is approximately within $2 \log R$ of the optimal, where R is the highest requirement value.

The worst-case guarantees for the heuristics of Goemans and Bertsimas and of Agrawal et al. were found by reduction to an ECON problem with costs satisfying the triangle inequality (see Step 1 and 4 of the tree heuristic). This reduction does not work, if the use of parallel edges is forbidden in the solution.

Recently, Khuller and Vishkin (1992) gave a heuristic for finding a minimum-cost k-edge connected subgraph of a graph (where parallel edges do not appear in the solution). This heuristic has a worst-case guarantee of 2, even when the costs do not satisfy the triangle inequality. Worst-case guarantees for heuristics for the NCON problem are not known.

5 Polynomially Solvable Special Cases

We have already remarked that the general problem of designing survivable networks is NP-hard; in fact, quite a number of special cases are known to be NP-hard. The intention of this section is to give an overview of those special cases where polynomial time solution procedures are known.

There are basically three ways to restrict the general model. One either considers special choices of node types, special cost functions or special classes of graphs. It turns out that some combinations of these restrictions lead to easy (but still interesting) problems.

5.1 Node Type Restrictions

Let us start by considering restrictions on the node types. $G = (V, E)$ may be any graph with costs $c_e \in \mathbf{R}$ for all $e \in E$.

If $r_v = 1$ for all $v \in V$ and $c_e > 0$ for all $e \in E$, the 1NCON and 1ECON problems for G and r are equivalent to finding a minimum cost spanning tree. This problem is well known to be solvable in polynomial time; see Chapter 12. If the costs are not necessarily positive, we look for a minimum cost connected subgraph. This can be found by first choosing all edges of nonpositive cost, shrinking all resulting components to single nodes, and computing a minimum spanning tree in the resulting graph.

If two nodes have type 1, say nodes u and v, all other nodes have type 0, and if all costs are positive, then the ECON (and NCON) problem for G and r is nothing but the problem of computing a shortest $[u, v]$-path; see Chapter 1 for polynomial time methods for that problem. If costs are general, we seek an edge set of minimum cost that contains a path from u to v. This can be solved in polynomial time by shrinking away the components induced by the nonpositive edges and then computing a shortest $[u, v]$-path in the resulting graph.

The "slightly" more general case, when $r_v \in \{0, 1\}$ for all $v \in V$, is the Steiner tree problem, which is NP-hard even if $c_e = 1$ for all $e \in E$. If there is a fixed number of nodes of type 0 or a fixed number of type 1, then the Steiner tree problem is polynomially solvable, see Lawler (1976).

The shortest path problem has an extension that is solvable in polynomial time. Namely, if two nodes have type k, say $r_u = r_v = k$, all others type 0, and if all costs are positive, then the kNCON problem asks for a collection of k node disjoint $[u, v]$-paths that are of minimum total cost and, similarly, the kECON problem requires finding a minimum collection of

k edge disjoint $[u,v]$-paths. Both problems can be solved with min-cost flow algorithms in polynomial time; see Chapter 1. As above, it is trivial to extend this to the case where costs are arbitrary.

We do not know of any other (nontrivial) case where special choices of node types lead to polynomial time solvability.

5.2 Cost Restrictions

The cases we want to mention here are two well-known problems in graph theory with uniform edge costs. First, given some nodes with node types, design a graph that satisfies all connectivity requirements and has a minimum number of edges; and second, given a graph and node types, add a minimum number of edges to the graph such that the resulting graph meets the connectivity requirements. Node and edge connectivity have to be distinguished here and whether or not parallel edges may be added.

Consider the following problem:

(5.1) Problem. *Given a set V of nodes, and a requirement $r_{st} \geq 0$, for each pair $s,t \in V$, find a graph $G=(V,E)$ such that each pair s,t is at least r_{st}-edge connected in G and such that $|E|$ is as small as possible.*

Chou and H. Frank (1970) gave a polynomial-time algorithm to solve this problem if G may contain parallel edges. H. Frank and Chou (1970) also solved this problem when no parallel edges are allowed, but additional nodes are allowed in the construction. If neither parallel edges nor further nodes are allowed, it is not known whether one can solve (5.1) in polynomial time.

The algorithm of Chou and Frank (1970) solves the ECON problem for K_n (with parallel edges), arbitrary edge connectivity requirements $r_{st} \geq 2$, and uniform edge costs.

The node connectivity problem analogous to (5.1) is open. We are only aware of a result of Harary (1962) who proved that, given n and k, the minimum number of edges in a k-connected graph on n nodes (without parallel edges) is $\lceil kn/2 \rceil$. Such a graph can be constructed easily.

To our knowledge these are the only solved cases with uniform edge costs.

The following problem often runs under the name **augmentation problem** in the graph theory literature.

(5.2) Problem. *Given a graph, augment it by a minimum number of edges so that the new graph meets certain connectivity requirements.*

In our terminology, (5.2) can be viewed as an ECON or NCON problem where all edges in the given graph have cost 0, and all edges that are allowed to be added have cost 1.

This type of problem was solved by Eswaran and Tarjan (1976) for 2-edge connected graphs, and by Rosenthal and Goldner (1977) for 2-node connected graphs. The k-edge connected graph augmentation problem was studied by Watanabe and Nakamura (1987), Ueno et al. (1988), Cai and Sun (1989), and Naor et al. (1990). Recently, A. Frank (1990) solved the general edge connectivity case. All solutions are algorithmic and require polynomial time.

A. Frank (1990) proved a nice min-max result for the minimum number of edges needed to augment a given graph G to satisfy given edge connectivity requirements r_{ij}. Let us define the **deficit** $\text{def}(A)$ of a node set A as

$$\text{def}(A) := \min_{u \in A, v \notin A} r_{uv} - |\delta_G(A)|.$$

The deficit of A is the smallest number of edges that have to be added to $\delta_G(A)$ in order to connect A sufficiently to all other nodes. Clearly, if several disjoint sets $A_i (i = 1, \ldots, t)$ have deficit $\text{def}(A_i)$, then a lower bound on the number of edges to be added is

$$\frac{1}{2} \sum_{i=1}^{t} \text{def}(A_i).$$

Frank (1990) shows that, under certain assumptions, the best such lower bound is exactly the minimum number of edges needed in an augmentation.

(5.3) Theorem. *Given a graph $G = (V, E)$ and edge connectivity requirements r_{st} for all pairs of nodes $s, t \in V$. Then the following holds:*

If all sets $A \subseteq V$ have deficit larger than 1, then the minimum number of edges to be added to G to satisfy the edge connectivity requirements, is

$$\max_{A_1, \ldots, A_t \, disjoint} \frac{1}{2} \sum_{i=1}^{t} \text{def}(A_i).$$

If some set A has deficit at most one and all proper subsets of A have nonpositive deficit, then the minimum number of edges to be added to G is def(A) plus the minimum number of edges to be added to $G - A$. $\square$

Frank's proof is constructive and results in polynomial-time algorithm to create a minimum cost augmentation. In the same paper, Frank proved similar results for directed graphs. But he did not consider augmentations to node-connected graphs, nor does he consider augmentations where the use of parallel edges is forbidden.

5.3 Special Classes of Graphs

It is often the case that NP-hard problems become easy when restricted to graphs with special structural properties. In the case of the ECON and NCON problems, there are only very few (and relatively simple) classes of graphs known where some versions can be solved in polynomial time. Most notable is the class of series-parallel graphs.

Series parallel graphs are created from a single edge by two operations:

- addition of parallel edges, and
- subdivision of edges.

(For our purposes we note that all series-parallel graphs, except K_2, are 2-connected.) The Steiner tree problem can be solved in linear time on this class of graphs by a recursive algorithm. This was mentioned by Takamizawa, Nishizeki and Saito (1982), and stated explicitly by Wald and Colbourn (1983). By a modification of this recursive algorithm, the 2NCON problem can also be solved, where node types 0, 1, and 2 are allowed.

Winter has developed linear-time algorithms for 2ECON and 2NCON problems with node types 0 and 2 on outerplanar, series-parallel, and Halin graphs, see Winter (1985a,b) and Winter (1986). In his survey article, Winter (1987) mentioned that he also found linear-time algorithms for Halin graphs that solve the 3ECON and 3NCON problems with node types 0 and 3.

Mahjoub (1988) has found a complete characterization of the polytope of 2-edge connected subgraphs of series-parallel graphs.

Complete characterizations of Steiner tree and related polytopes in extended spaces can be found in Prodon, Liebling, and Gröflin (1985) and Goemans (1991a, 1991b). Cornuéjols, Fonlupt, and Naddef (1985) found a complete description of the dominant of the 2-edge connected subgraph polytope of series parallel graphs, and Chopra (1991) investigated, for odd k, the k-edge connected subgraphs of a given outerplanar graph, and found a complete description of the dominant of the associated polyhedron by the so-called **lifted outerplanar partition inequalities**. (Outerplanar graphs are those graphs that can be drawn in the plane as a cycle with non-crossing chords. They form a subclass of the class of series-parallel graphs.) Chopra's result is a follows:

(5.4) Theorem. *For outerplanar graphs $G = (V, E)$ and uniform node types $r \in \{k\}^V$, k odd, the dominant of the $kECON(G;r)$ polytope is completely characterized by the inequalities:*

$$\begin{array}{ll} \frac{1}{2}\sum_{i=1}^{p} x(\delta(W_i)) \geq p \cdot \lceil k/2 \rceil - 1 & \text{for all partitions } \{W_1, \ldots, W_p\} \text{ of } V_j \\ x_e \geq 0 & \text{for all } e \text{ in } E. \ \Box \end{array}$$

Note, however, that these inequalities are not generally valid for $kECON(G;r)$ if the underlying graph is, say, complete!

6 Polyhedral Results

Except for the results of Grötschel & Monma (1990) mentioned in Section 3, there is not much known about the polytope CON$(G; r, k, d)$ for general edge and node survivability requirements r, k and d. We will thus concentrate on the kNCON and kECON problems that have been investigated in more depth and survey some of the known results. Particular attention has been paid to the low-connectivity case, that is, where $r \in \{0, 1, 2\}^E$. See Grötschel & Padberg (1985) and Pulleyblank (1990) for a general survey of polyhedral combinatorics and the basics of polyhedral theory.

Let us mention again the idea behind this approach and its goal. We consider an integer programming problem like (3.1) or (3.10). We want to turn such an integer program into a linear program and solve it using the (quite advanced) techniques of this area. To do this, we define a polytope associated with the problem by taking the convex hull of the feasible (integral) solutions of a program like (3.1) or (3.10). Let P be such a convex hull. We know from linear programming theory that, for any objective function c, the linear program $\min\{c^T x \mid x \in P\}$ has an optimum vertex solution (if it has a solution). This vertex solution is, by definition, a feasible solution of the initial integer program and thus, by construction, an optimum solution of this program.

The difficulty with this approach is that max $c^T x$, $x \in P$ is a linear program only "in principle". To provide an instance to an LP-solver, we have to find a different description of P. The polytope P is defined as the convex hull of (usually many) points in $\mathbf{R}^E$, but we need a complete (linear) descriptions of P by means of linear equations or inequalities. The Weyl-Minkowski theorem tells us that both descriptions are in a sense equivalent, in fact, there are constructive procedures that compute one description of P from the other. However, these procedures are inherently exponential and nobody knows how to make effective use of them, in particular, for NP-hard problem classes. Moreover, there are results in complexity theory, see Papadimitriou and Yannakakis (1982), that indicate that it might be much harder to find a complete linear description of such a polytope P than to solve $\min c^T x,\ x \in P$.

At present, no effective general techniques are known for finding complete or "good partial" descriptions of such a polytope or large classes of facets. There are a few basic techniques like the derivation of so-called Chvátal cuts (see Chvátal (1973)). But most of the work is a kind of "art". Valid inequalities are derived from structural insights and the proofs that many of these inequalities define facets use technically complicated, ad-hoc arguments.

If large classes of facet-defining inequalities are found, one has to think about their algorithmic use. The standard technique is to employ such inequalities in the framework of a cutting plane algorithm. We will explain this in Section 7. It has turned out in the recent years that such efforts seem worthwhile. If one wants to find true optimum solutions or extremely good lower bounds, the methods of polyhedral combinatorics are the route to take.

6.1 Classes of Valid Inequalities

We will now give an overview of some of the results of Grötschel, Monma and Stoer (1992a, 1992b, 1992c) and Stoer (1992) concerning classes of valid inequalities for the kECON and kNCON problems. We will motivate these inequalities and mention how they arise.

As before, we consider a loopless graph $G = (V, E)$, and in the kECON case possibly with multiple edges. We assume that for each node $v \in V$ a nonnegative integer r_v, its node type, is given, that $k = \max\{r_v \mid v \in V\}$ and that at least two nodes are of type k. Recall that $r(W) = \max\{r_v \mid v \in W\}$ is called the node type of W.

We start out by repeating those classes we have already introduced in Section 3. Clearly, the **trivial inequalities**

$$(6.1) \qquad 0 \le x_e \le 1 \qquad \text{for all } e \in E$$

are valid for $k\text{ECON}(G; r)$ and $k\text{NCON}(G; r)$ since problem (3.10) is a 0/1-optimization problem. The **cut inequalities**

$$(6.2) \qquad x(\delta(W)) \ge \text{con}(W) \qquad \text{for all } W \subseteq V, \emptyset \ne W \ne V,$$

where $\text{con}(W) = \min\{r(W),\ r(V \setminus W)\}$, are valid for $k\text{ECON}(G; r)$ and $k\text{NCON}(G; r)$ since the survivable network to be designed has to contain at least $\text{con}(W)$ edge disjoint paths that connect nodes in W to nodes in $V \setminus W$. (Recall that $r_{st} = \min\{r_s, r_t\}, s, t \in V$.) In the node connectivity case the requirement that if, for a given pair s, t of nodes, at most $r_{st} - 1$ nodes different from s and t are deleted from G then there has to be at least one more path connecting s and t in the remaining graph. This requirement leads to the **node cut inequalities.**

$$\text{(6.3)} \quad x(\delta_{G-Z}(W)) \geq \mathrm{con}(W) - |Z| \qquad \begin{array}{l} \text{for all pairs } s, t \in V, s \neq t \text{ and} \\ \text{for all } \emptyset \neq Z \subseteq V \setminus \{s, t\} \text{ with } |Z| \leq r_{st} - 1 \text{ and} \\ \text{for all } W \subseteq V \setminus Z \text{ with } s \in W, t \notin W. \end{array}$$

These inequalities are valid for kNCON$(G; r)$ but – of course – not for kECON$(G; r)$.

How does one find further classes of valid inequalities? One approach is to infer inequalities from structural investigations. For instance, the cut inequalities ensure that every cut separating two nodes contains at least r_{st} edges. These correspond to partitioning the node set into two parts and guaranteeing that there are enough edges linking them. We can generalize this idea as follows. Let us call a system $W_1, \ldots, W_p$ of subsets of V with $W_i \cap W_j = \emptyset$ for $1 \leq i < j \leq p$, $W_i \neq \emptyset$ for $i = 1, \ldots, p$ and $W_1 \cup \ldots \cup W_p = V$ a **partition** of V and let us call

$$\delta(W_1, \ldots, W_p) := \{uv \in E \mid \exists i, j, 1 \leq i, j \leq p, i \neq j \text{ with } u \in W_i, v \in W_j\}$$

a **multicut** or p-**cut** (if we want to specify the number p of **shores** $W_1, \ldots, W_p$ of the multicut). Depending on the types $r(W_1), \ldots, r(W_p)$, any survivable network (V, F) will have to contain at least a certain number of edges of the multicut $\delta(W_1, \ldots, W_p)$. For every partition it is possible to compute a lower bound of this number, and thus to derive a valid inequality for every node partition (resp. multicut). This goes as follows.

Suppose $W_1, \ldots, W_p$ is a partition of V such that $r(W_i) \geq 1$ for $i = 1, \ldots, p$. Let $I_1 := \{i \in \{1, \ldots p\} \mid r(W_i) = 1\}$, $I_2 := \{i \in \{1, \ldots, p\} \mid r(W_i) \geq 2\}$. Then the **partition inequality** (or multicut inequality) induced by $W_1, \ldots W_p$ is defined as

$$\text{(6.4)} \qquad x(\delta(W_1, \ldots, W_p)) = \frac{1}{2} \sum_{i=1}^{p} x(\delta(W_i)) \geq \begin{cases} \lceil \frac{1}{2} \sum\limits_{i \in I_2} r(W_i) \rceil + |I_1| & \text{if } I_2 \neq \emptyset \\ p - 1 & \text{if } I_2 = \emptyset \end{cases}$$

Every partition inequality is valid for kECON$(G; r)$ and thus for kNCON$(G; r)$.

Structural insight of this type is often hard to get without any "hint". We mention a second approach that frequently helps to get a fresh view. One solves a number of linear programs using various objective functions and all inequalities and equations which are currently known. In some cases, fractional optimum solution will result. The question is now to invent new inequalities that cut these fractional solutions off but do not cut off any integral solution. If one can find such inequalities then the research challenge is to produce a large class of more general valid inequalities that contain the initial ones as special cases. The hint towards establishing new types of valid inequalities is given by the structure of the fractional optimum solutions. Again there is no general technique known for effectively using this information. Some creativity is necessary.

For example, consider a complete graph $K_3 = (V, E)$ on the three nodes $V = \{a, b, c\}$, the node types are $r_v = 1$ for all $v \in V$. The polyhedron defined by the trivial inequalities (6.1) and the cut inequalities (6.2) has the fractional vertex $x_e = \frac{1}{2}$ for all $e \in E$. This fractional solution does not satisfy the partition inequality $x(\delta(\{a\}, \{b\}, \{c\})) = x_{ab} + x_{ac} + x_{bc} \geq 2$.

Just as the cut inequalities (6.2) can be generalized as outlined above to partition inequalities (6.4), the node cut inequalities (6.3) can be generalized to a class of inequalities that we will call node partition inequalities, as follows.

Let $Z \subseteq V$ be some node set with $|Z| \geq 1$. If we delete Z from G then the resulting graph must contain an $[s, t]$-path for every pair of nodes s, t of type larger than $|Z|$. In other words, if $W_1, \ldots, W_p$ is a partition of $V \setminus Z$ into node sets with $r(W_i) \geq |Z| + 1$ then the graph $G' := (G - Z)/W_1/\ldots/W_p$ obtained by deleting Z and contracting $W_1, W_2, \ldots,$ and W_p must be connected. This observation gives the following class of **node partition inequalities** valid for kNCON$(G; r)$, but not for kECON$(G; r)$:

(6.5) $\quad \frac{1}{2} \sum_{i=1}^{p} x(\delta_{G-Z}(W_i)) \geq p - 1$ $\quad$ for every node set $Z \subseteq V, |Z| \geq 1$ and every partition $W_1, \ldots, W_p$ of $V \setminus Z$ such that $r(W_i) \geq |Z| + 1, i = 1, \ldots, p.$

If $r(W_i) \geq |Z| + 2$ for at least two node sets in the partition, then the right-hand side of the node partition inequality can be increased. This leads to further generalizations of the classes (6.4) and (6.5), but their description is quite technical and complicated, see Stoer (1992). So we do not discuss them here.

We now mention another approach (that can be viewed as a special case of our first one) to finding new classes of valid inequalities.

The idea here is to relax the problem in question combinatorially by finding a (hopefully easier) further combinatorial optimization problem such that every solution of the given problem is feasible for the new problem and by studying the polytope associated with the new combinatorial optimization problem. If the relaxation is carefully chosen (and one is lucky) some valid inequalities for the relaxed polytope turn out to be facet-defining for the polytope one wants to consider. (These inequalities are trivially valid.)

In our case, a relaxation that is self-suggesting is to consider the r-cover problem. This ties the survivability problem to matching theory and, in fact, one can make good use of the results of this theory for the survivability problem.

The survivability requirements imply that if $v \in V$ is a node of type r_v, then v has degree at least r_v for any feasible solution of the kECON problem. Thus, if we can find an edge set such that each node has degree at least r_v (we call such a set an **r-cover**) and that has minimum cost we obtain a lower bound for the optimum value of the kECON problem. Clearly, such an edge set can be found by solving the integer linear program

$$\begin{array}{llll} (6.6) & & \min \mathrm{c}^{\mathrm{T}}\mathrm{x} & \\ & \text{(i)} & x(\delta(v)) \geq r_v & \text{for all } v \in V, \\ & \text{(ii)} & 0 \leq x_e \leq 1 & \text{for all } e \in E, \text{and} \\ & \text{(iii)} & x_e \text{ integer} & \text{for all } e \in E, \end{array}$$

which is obtained from (3.1) (i), (iii), (iv) by considering only sets of cardinality one in (3.1) (i). The inequalities (i) are called **degree constraints**. This integer program can be turned into a linear program, i.e., the integrality constraints (iii) are replaced by a system of linear inequalities, using Edmonds' polyhedral results on b-matching, see Edmonds (1965). Edmonds proved that, for any vector $b \in Z_+^V$, the vertices of the polyhedron defined by

$$\begin{array}{llll} (6.7) & \text{(i)} & y(E(H)) + y(\bar{T}) \leq \lfloor \sum\limits_{v \in H} (b_v + |\bar{T}|)/2 \rfloor & \text{for all } W \subseteq V \text{and all } \bar{T} \subseteq \delta(H), \text{and} \\ & \text{(ii)} & 0 \leq y_e \leq 1 & \text{for all } e \in E \end{array}$$

are precisely the incidence vectors of all (1-capacitated) **b-matchings** of G, i.e., of edge sets M such that no node $v \in V$ is contained in more than b_v edges of M. For the case $b_v := \deg(v) - r_v$, where $\deg(v)$ denotes the degree of v in G, the b-matchings M are nothing but the complements $M = E \setminus F$ of r-covers F of G. Using the transformation $x := \mathbf{1} - y$ and $T := \delta(H) \setminus \bar{T}$ we obtain the system

$$\begin{array}{lll} (6.8) & \text{(i) } x(E(H)) + x(\delta(H) \setminus T) \geq \lfloor \sum\limits_{v \in H} (r_v - |T|)/2 \rfloor & \text{for all } H \subseteq V \text{and all } T \subseteq \delta(H), \text{and} \\ & \text{(ii) } 0 \leq x_e \leq 1 & \text{for all } e \in E. \end{array}$$

(6.8) gives a complete description of the convex hull of the incidence vectors of all r-covers of G. We call the inequalities (6.8)(i) ***r*-cover inequalities**. Since every solution of the kECON problem for G and r is an r-cover, all inequalities (6.8)(i) are valid for $k\mathrm{ECON}(G;r)$. It is a trivial matter to observe that those inequalities (6.8)(i) where $\sum_{v \in H} r_v - |T|$ is even are redundant. For the case $r_v = 2$ for all $v \in V$, Mahjoub (1988) described the class of r-cover inequalities, which he calls odd wheel inequalities.

Based on these observations one can extend inequalities (6.8)(i) to more general classes of inequalities valid for $k\mathrm{ECON}(G;r)$ (but possibly not valid for the r-cover polytope). We present here one such generalization.

Let H be a subset of V called the **handle**, and $T \subseteq \delta(H)$ with $|T|$ odd and $|T| \geq 3$. For each $e \in T$, let T_e denote the set of the two endnodes of e. The sets $T_e, e \in T$, are called **teeth**. Let $H_1, \ldots, H_p$ be a partition of H into nonempty disjoint subsets such that $r(H_i) \geq 1$ for $i = 1, \ldots, p$ and $|H_i \cap T_e| \leq r(H_i) - 1$ for all $i \in \{1, \ldots, p\}$ and all $e \in T$. Let $I_1 := \{i \in \{1, \ldots, p\} \mid r(H_i) = 1\}$ and $I_2 = \{i \in \{1, \ldots, p\} \mid r(H_i) \geq 2\}$. We call

$$\begin{aligned} x(E(H)) - \sum_{i=1}^{p} x(E(H_i)) + x(\delta(H) \setminus T) \\ \geq \Big\lceil \sum_{i \in I_2} (r(H_i) - |T|)/2 \Big\rceil + |I_1| \end{aligned} \tag{6.9}$$

the **lifted r-cover inequality** (induced by $H_1, \ldots, H_p, T$). All inequalities of type (6.9) are valid for $\mathrm{ECON}(G;r)$.

The names "handle" and "teeth" used above derive from the observation that there is some relationship of these types of inequalities with the 2-matching, comb and clique tree inequalities for the symmetric traveling salesman polytope; see Chapter 3. In fact, there is a further generalization of the r-cover inequalities that makes this resemblance (in form and name) more apparent. This generalization, however, applies only to the case where $r_v \in \{0, 1, 2\}$ for all $v \in V$.

Let $H, T_1, \ldots, T_t$ be subsets of V (H is the handle, the sets $T_1, \ldots, T_t$ are the teeth) that satisfy the following conditions. The number t of teeth is at least 3 and odd. Two teeth have at most one node in common. Each tooth T_i intersects the handle H in exactly one node; we denote this node by t_i for $i = 1, \ldots, t$. In each tooth T_i we choose a node $z_i \in T_i \setminus H$ and call z_i a **special node**. The choice is restricted if $T_i \cap T_j \neq \emptyset$; in this case $T_i \cap T_j = \{z_i\} = \{z_j\}$ (i.e., the nodes z_i are not necessarily distinct). Moreover, we assume that $r_{t_i} = 2$ for $i = 1, \ldots, t$ and $r_v \geq 1$ for all $v \in H \cup (\bigcup_{i=1}^{t} (T_i \setminus \{z_i\}))$. Under these assumptions one can show that the following **comb inequality**

$$\begin{aligned} x(E(H)) + x(\delta(H)) + \sum_{i=1}^{t} x(E(T_i)) + \sum_{i=1}^{t} x([T_i \setminus (H \cup \{z_i\}) : V \setminus T_i]) \\ - \sum_{i=1}^{t} x([\{t_i\} : T_i]) - \sum_{i=1}^{t} x([\{z_i\} : T_i \cap V_2]) \geq |H| + \sum_{i=1}^{t} (|T_i| - 2) - \lfloor \frac{t}{2} \rfloor \end{aligned} \tag{6.10}$$

is valid for $\mathrm{2NCON}(G;r)$; it is not valid for $\mathrm{2ECON}(G;r)$. Again there are further generalizations that we do not want to discuss here.

Another technique for finding further classes of valid and facet-defining inequalities will be mentioned in Section 6.4.

6.2 Facet Results

To develop successful cutting plane algorithms, it is not enough to know some inequalities valid for the polytope over which one wants to optimize. The classes of inequalities should contain large numbers of facets of the polytope. Ideally, one would like to use classes of facet-defining inequalities only.

In our case, it turned out to be extremely complicated to give (checkable) necessary and sufficient conditions for an inequality in one of the classes described above to define a facet of NCON($G; r$) or ECON($G; r$). Lots of technicalities creep in. Nevertheless, it could be shown that large subsets of these classes are facet-defining. These results provide a theoretical justification for the use of these inequalities in a cutting plane algorithm.

We will not go deeply into the results that have been obtained so far; see Grötschel and Monma (1990), Grötschel, Monma and Stoer (1992a,1992b,1992c) and Stoer (1992) for details. We just provide a glimpse at the subject here.

We restrict ourselves to the cut inequalities (6.2). We first concentrate on the low-connectivity case, where $r_v \in \{0,1,2\}$ for all $v \in V$. To state the results we have to introduce further terminology.

Given a graph $G = (V, E)$, a node type vector $r \in \mathbf{Z}_+^V$, and a node set $W \subseteq V$ with $|W| \geq 2$, we set

V_i	$:=$	$\{v \in V \mid r_v \geq i\}$ for $i = 0, 1, 2$,
$\lambda(G, W)$	$:=$	minimum cardinality of a subset of E whose removal from G disconnects two nodes of W, and
$\kappa(G, W)$	$:=$	minimum cardinality of a set $S \subseteq V \cup E$ whose removal from G disconnects two nodes of W in G (recall that we assume G to be simple in this case).

If $|W| < 2$ then $\lambda(G, W)$ and $\kappa(G, W)$ are defined to be ∞.

We assume throughout this section that G is two-node connected and satisfies $\lambda(G, V_2) \geq 3$ when we deal with the 2ECON case, and that G is two-node connected and satisfies $\kappa(G, V_2) \geq 3$ when we deal with the 2NCON case. If this is not so then the problem can be decomposed (in polynomial time) into independent smaller problems that are trivially solvable or satisfy these conditions. The exact decomposition procedure involves many technical details and is described in Grötschel, Monma and Stoer (1992b). A side benefit of this assumption is that the polyhedra 2ECON($G; r$) and 2NCON($G; r$) are full dimensional (see Grötschel, Monma (1990)), i.e., they contain $|E| + 1$ affinely independent vectors.

The next theorem describes necessary conditions for the cut inequalities (6.2) to define facets.

(6.11) Theorem. *Let $G = (V, E)$ be a graph, $r \in \{0, 1, 2\}^V$ and $W \subseteq V$ with $\emptyset \neq W \neq V$.*

(i) *If $r(W) = 2$ then $x(\delta(W)) \geq 2$ defines a facet of* 2ECON$(G; r)$ *only if $\lambda(G[W], V_1) \geq 2$, $\lambda(G[V \setminus W], V_1) \geq 2$, and $G[W]$ and $G[V \setminus W]$ are connected.*

(ii) *If $r(W) = 1$ then $x(\delta(W)) \geq 1$ defines a facet of* 2ECON$(G; r)$ *if and only if* $G[W]$ and *$G[V \setminus W]$ are connected, $\lambda(G[W], V_i) \geq i + 1$ for $i = 1, 2$, and $\lambda(G[V \setminus W], V_2) \geq 3$.*

(iii) *If $r(W) = 0$ then $x(\delta(W)) \geq 0$ does not define a facet of* 2ECON$(G; r)$ *or* 2NCON *$(G; r)$.*

(iv) *If $r(W) = 2$ then $x(\delta(W)) \geq 2$ defines a facet of* 2NCON$(G; r)$ *only if all conditions of (i) are satisfied and $\kappa(G[W], V_2) \geq 2$ and $\kappa(G[V \setminus W], V_2) \geq 2$.*

(v) *If $r(W) = 1$ then $x(\delta(W)) \geq 1$ defines a facet of* 2NCON$(G; r)$ *only if all conditions of (ii) are satisfied and $\kappa(G[V \setminus W] - e, V_2) \geq 2$ for all $e \in E(V \setminus W)$.* $\square$

If we look at the uniform case where $r_v = 2$ for all $v \in V$, then in a 3-edge connected graph a cut inequality $x(\delta(W)) \geq 2$ defines a facet of 2ECON$(G; r)$ if and only if $G[W]$ and $G[V \setminus W]$ are both 2-edge connected. This result was independently obtained in Mahjoub (1988).

Results (i), (iv) and (v) above can be turned into "if and only if" statements by adding further (complicated) technical conditions, see Grötschel, Monma and Stoer (1992a). Things get even more complicated if we want to characterize those node cut inequalities (6.3) that are facet-defining for 2NCON$(G; r)$; see Stoer (1992). For partition, node partition, r-cover, lifted r-cover and comb inequalities to define a facet only sufficient conditions are known (in the low- as well as in the high- connectivity cases).

If r is a vector of arbitrary nonnegative integers, even more complications arise due to the fact that the structure of the graph and the distribution of the node types interact and heavily restrict the structure of the feasible solutions. We can offer only one result that provides necessary and sufficient conditions. It is, again, for the structurally rather simple cut inequalities and for the edge-connectivity case. Moreover, we have to assume that the given graph is highly connected and all nodes have the same type. This result is due to Stoer (1992).

(6.12) Theorem. *Let $G = (V, E)$ be a $(k+1)$-edge connected graph, let $r_v = k$ for all nodes $v \in V$, and let $W \neq V$ be a nonempty node set. Define for each $W_i \subseteq W$ with $\emptyset \neq W_i \neq W$ the deficit of W_i as*

$$\mathrm{def}_G(W_i) := \max\{0, k - |\delta_{G[W]}(W_i)|\}.$$

Define similarly for $U_i \subseteq V \setminus W$ with $\emptyset \neq U_i \neq V \setminus W$

$$\mathrm{def}_G(U_i) := \max\{0, k - |\delta_{G[V \setminus W]}(U_i)|\}.$$

The cut inequality

$$x(\delta(W)) \geq k$$

defines a facet of the polytope kECON$(G;r)$ *of k-edge connected graphs if and only if*

(a) $G[W]$ *and* $G[V \setminus W]$ *are connected, and*

(b) *for all edges* $e \in E(W) \cup E(V \setminus W)$,
for all pairwise disjoint node sets $W_1, \ldots, W_p$ *of* W*, with* $W_i \neq W$ *for all* $i = 1, \ldots, p$ $(p \geq 0)$*, and*
for all pairwise disjoint node sets $U_1, \ldots, U_q$ *of* $V \setminus W$*, with* $U_i \neq V \setminus W$ *for all* $i = 1, \ldots, q$ $(q \geq 0)$,
the following inequality holds:

$$\sum_{i=1}^{p} \mathrm{def}_{G-e}(W_i) + \sum_{i=1}^{q} \mathrm{def}_{G-e}(U_i) - |[\bigcup_{i=1}^{p} W_i : \bigcup_{i=1}^{q} U_i]| \leq k. \quad \square$$

Summarizing, we can say that, in the general kECON or kNCON case, facet characterizations are technically rather difficult and that in many cases it seems quite hopeless to get (reasonable) necessary and sufficient conditions for an inequality (of one of the classes introduced above) to define a facet of kECON$(G;r)$ or kNCON$(G;r)$. Nevertheless, the results established so far (i.e., the sufficient conditions) show that these classes do contain many facet-defining inequalities.

6.3 Separation

Note that – except for the trivial inequalities – all classes of valid inequalities for the kECON and kNCON problem described in Section 6.1 contain a number of inequalities that is exponential in the number of nodes of the given graph. So it is impossible to input these inequalities into an LP-solver. But there is an alternative approach. Instead of solving an LP with all inequalities, we solve one with a few "carefully selected" inequalities and we generate new inequalities as we need them. This approach is called a cutting plane algorithm and works as follows.

We start with an initial linear program. In our case, it consists of the linear program (6.6) without the integrality constraints (iii). We solve this LP. If the optimum solution y is feasible for the kECON or kNCON problem, then we are done. Otherwise we have to find some inequalities that are valid for kECON$(G;r)$ or kNCON$(G;r)$ but are violated by y. We add these inequalities to the current LP and repeat.

The main difficulty of this approach is in efficiently generating violated inequalities. We state this task formally.

(6.13) Separation Problem (for a class C of inequalities)

Given a vector y decide whether y satisfies all inequalities in C and, if not, output an inequality violated by y.

A trivial way to solve (6.13) is to substitute y into each of the inequalities in C and check whether one of the inequalities is violated. But in our case this is too time consuming since C is of size exponential in $|V|$. Note that all the classes C described before have an implicit description by means of a formula with which all inequalities can be generated. It thus may happen that algorithms can be designed that check violation much more efficiently than the trivial substitution process. We call an algorithm that solves (6.13) an (exact) **separation algorithm** for C, and we say that it runs in polynomial time if its running time is bounded by a polynomial in $|V|$ and the encoding length of y.

A deep result of the theory of linear programming (see Grötschel, Lovász, Schrijver (1988)) states (roughly) that a linear program over a class C of inequalities can be solved in polynomial time if and only if the separation problem for C can be solved in polynomial time. Being able to solve the separation problem thus has considerable theoretical consequences.

This result makes use of the ellipsoid method and does not imply the existence of a "practically efficient" algorithm. However, by combining separation algorithms with other LP solvers (like the simplex algorithms) can result in quite successful cutting plane algorithms; see Section 7.

Our task now is to find out whether reasonable separation algorithms can be designed for any of the classes (6.2), (6.3), (6.4), (6.5), (6.8)(i), (6.9) or (6.10).

There is some good and some bad news. The good news is that for the cut inequalities (6.2), the node cut inequalities (6.3) and the r-cover inequalities (6.8)(i), exact separation algorithms are known that run in polynomial time; see Grötschel, Monma and Stoer (1992c).

Let us consider the class C of cut inequalities (6.2), let y be some vector in $\mathbf{R}^E$ with $0 \leq y_e \leq 1$ for all $e \in E$ and let r be in $\mathbf{Z}_+^V$. We view the components y_e as capacities of the edges of the given graph $G = (V, E)$ and compute a Gomory-Hu tree (using the Gomory-Hu algorithm that consists of $|V| - 1$ calls of a max-flow algorithm and some bookkeeping; see Gomory-Hu (1961)). The Gomory-Hu tree has the property that, for any two nodes $u, v \in V$, the minimum capacity of a cut separating u and v is given by the smallest weight of an edge that is contained in the unique path linking u and v in the Gomory-Hu tree. Having the Gomory-Hu tree T with weights w_e for all $e \in T$ we can determine whether y satisfies all cut inequalities as follows. For each edge e, let V_e and V_e' denote the node sets of the two components of $(V, T-e)$. If there is an edge $e \in E$ with

$$w_e < \min\{\max\{r_v \mid v \in V_e\}, \max\{r_v | v \in V'_e\}\}$$

then y violates the inequality $x(\delta(V_e)) \geq r(W)$, otherwise y satisfies all cut inequalities $x(\delta(W)) \geq r(W)$. Hao & Orlin (1992) describe a faster algorithm that finds a minimum cut in a graph without finding a Gomory-Hu tree, and whose running time is equal to the running time of one maxflow routine. The separation problem for the node cut constraints (6.3) can be reduced to a sequence of minimum (s,t)-cut computations in a directed graph. This polynomial time method is described in Grötschel, Monma & Stoer (1992c).

The polynomial time exact separation algorithm for the r-cover inequalities is based on the Padberg-Rao procedure for solving the separation problem for the capacitated b-matching inequalities; see Padberg and Rao (1982). The "trick" is to reverse the transformation from the b-matching to the r-cover problem described in (6.7) and (6.8) and call the Padberg-Rao algorithm. It is easy to see that y satisfies all r-cover inequalities (6.8)(i) if and only if its transformation satisfies all b-matching inequalities (6.7)(i). The Padberg-Rao procedure is quite complicated to describe, so we do not discuss it here.

The bad news is that it was shown in Grötschel, Monma and Stoer (1992b) that the separation problems for partition inequalities (6.4), node partition inequalities (6.5) and lifted r-cover inequalities (6.9) are NP-hard. (See Section 8 for a way to cure this problem in some cases by using directed models and projections.)

Thus, in these cases we have to revert to **separation heuristics**, i.e., fast procedures that check whether they can find an inequality in the given class that is violated by y, but which are not guaranteed to find one even if one exists. We discuss separation heuristics in more detail in Section 7.

6.4 Complete Descriptions of Small Cases

For the investigation of combinatorially-defined polyhedra, it is often useful to study small cases first. A detailed examination of such examples provides insight into the relationship between such polyhedra and gives rise to conjectures about general properties of these polyhedra. Quite frequently a certain inequality is found, by numerical computation, to define a facet of a kECON or kNCON polytope of small dimension. Afterwards it is often possible to come up with a class (or several classes) of inequalities that generalize the given one and to prove that many inequalities of these classes define facets of combinatorial polytopes of any dimension.

By means of a computer program, we have computed complete descriptions of all kECON and kNCON polytopes of small dimensions. To give a glimpse of these numerically obtained results, we report here the complete descriptions of all 2ECON and all 2NCON polytopes of the complete graphs on five vertices K_5. More information about small kECON polytopes can be found in Stoer (1992).

6.4.1 The 2ECON Polytope for K_5

Let us begin with the polytopes 2ECON (K_5, r) where $r = (r_1, \ldots, r_5)$ is the vector of node types. The node types r_i have value 0, 1 or 2, and by assumption, at least two nodes are of highest type 2. Clearly, we can suppose that $r_i \geq r_{i+1}$ for $i = 1, \ldots, 4$. These assumptions result in ten node type vectors to be considered. It is obvious that, if a node type vector r componentwise dominates a vector r' (i.e., $r_i \geq r'_i$ for all i) then 2ECON$(K_n, r) \subseteq$ 2ECON(K_n, r').

Figure 6.1 provides a comprehensive summary of our findings. In this figure, a polytope 2ECON(K_5, r) is depicted by its node type vector $r = (r_1, \ldots, r_5)$. A line linking two such vectors indicates that the polytope at the lower end of the line directly contains the polytope at the upper end of the line and that no other 2ECON polytope is "in between". For example, the polytope 2ECON$(K_5, (2,2,2,1,0))$ is directly contained in 2ECON$(K_5, (2,2,1,1,0))$ and 2ECON$(K_5(2,2,2,0,0))$, and it contains directly 2ECON$(K_5, (2,2,2,1,1))$ and 2ECON $(K_5(2,2,2,2,0))$. Next to the right or left of a node type vector r, a box indicates which inequalities are needed to define the corresponding 2ECON(K_5, r) polytope completely and nonredundantly. The notation is as follows:

- The type of inequality appears in the first column.
  - "p" stands for "partition inequality", see (6.4),
  - "cut" stands for "cut constraint", see (6.2),
  - "rc" stands for "lifted r-cover inequality", see (6.9),
  - "d" stands for "degree constraint", see (6.6)(i),
  - "rc+1", "I1", and "I2", stand for new types of inequalities explained later.

- The next column lists, for each partition of the handle ("rc") or the whole node set ("p"), the node types in each node set. The different sets are separated by commas.
  - For instance, "p 200,2,1" stands for a partition inequality induced by a partition of V, whose first node set contains one node of type 2 and two nodes of type 0,

whose second set contains exactly one node of type 2, and whose last set contains exactly one node of type 1.

- "rc 20,2,2" stands for a lifted r-cover inequality induced by a handle that is partitioned into three node sets, the first one containing two nodes, a node of type 2 and a node of type 0, the second node set containing exactly one node of type 2, and the third node set containing exactly one node of type 2; the number of teeth can be computed with the help of the right-hand side.

- The next column gives the right-hand side.
- The last column contains the number of inequalities of the given type.

We do not list the trivial inequalities $0 \leq x_e \leq 1$. We know that $x_e \leq 1$ always defines a facet of the considered polytopes, and that $x_e \geq 0$ defines a facet if and only if e is not contained in a cutset $\delta(W)$ of size $\mathrm{con}(W) + 1$.

The inequalities denoted by "rc+1", "I1", and "I2" in the polytopes $\mathrm{2ECON}(K_5; (2\ 2\ 2\ 0\ 0))$, and $\mathrm{2ECON}(K_5; (2\ 2\ 2\ 1\ 0))$, are depicted in Figure 6.2. All except I2 have coefficients in $\{0, 1, 2\}$. The coefficients of inequality I2 in Figure 6.2 take values in $\{0, 1, 2, 3\}$. Edges of coefficient 0 are not drawn, edges of coefficient 1 are drawn by thin lines, and edges of coefficient 2 are drawn by bold lines. To make this distinction somewhat clearer, we additionally display the coefficients of all thin or of all bold lines.

This numerical study of 2ECON polytopes of K_5 reveals that degree, cut, partition and lifted r-cover inequalities play a major role in describing the polytopes completely and nonredundantly. But at the same time we found three new classes of (complicated) inequalities that are presently the subject of further studies.

6.4.2 The 2NCON Polytope for K_5

We now turn our attention to the 2NCON polytopes of the complete graph K_5. It turned out that only two further classes of inequalities are needed to describe all polytopes 2NCON (K_5, r) completely and nonredundantly. These are the classes of node cut and node partition inequalities (6.3) and (6.5). Figure 6.3 displays the complete descriptions of the 2NCON polytopes for K_5 in the same way as Figure 6.1 does for the 2ECON polytopes. The (new) entries for the node cut and node partition inequalities read as follows.

"ncut 20,20" denotes a node cut inequality $x(\delta_{G-z}(W))$, where W contains a node of type 2 and a node of type 0, and $V \backslash (W \cup \{z\})$ contains a node of type 2 and a node of type 0; the

"." in "ncut 2.,2." represents a node of any type; and "np 20,2,2" denotes a node partition inequality induced by a partition of $V \backslash \{z\}$, where z is some node in V, and the first shore of the partition consists of a node of type 2 and a node of type 0, the second shore consists of a node of type 2, and the third shore consists of a node of type 2.

This concludes our examination of polyhedra for small instances of 2NCON and 2ECON. For further such results, see Stoer (1992).

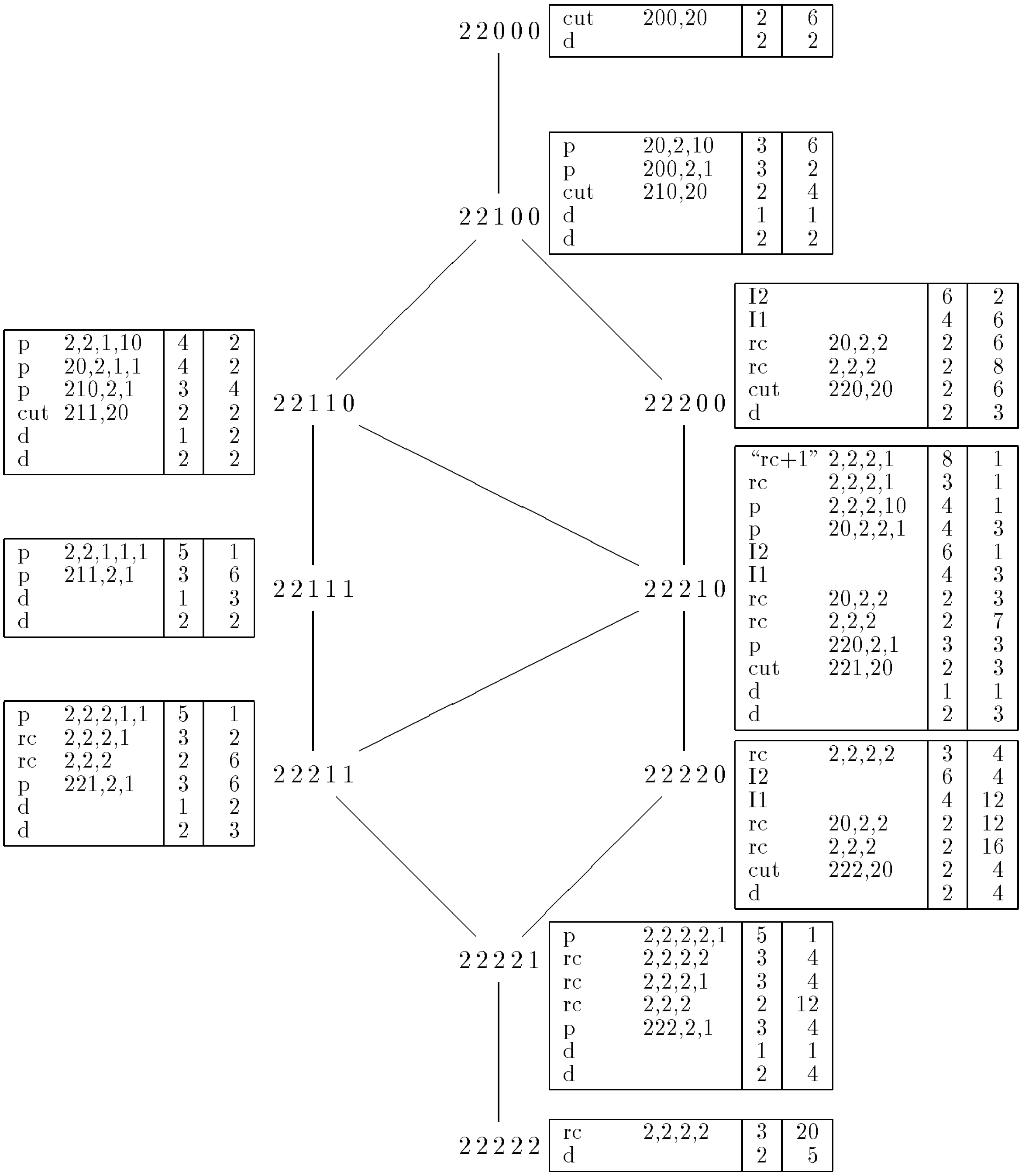


Figure 6.1: 2ECON($K_5; r$) polyhedra

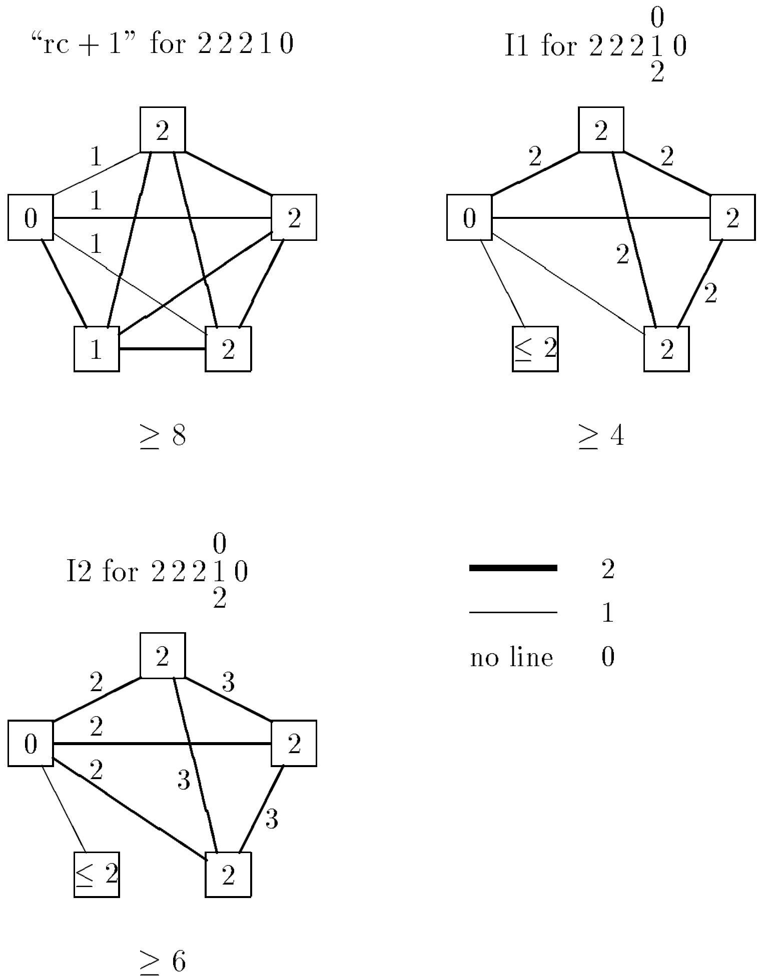
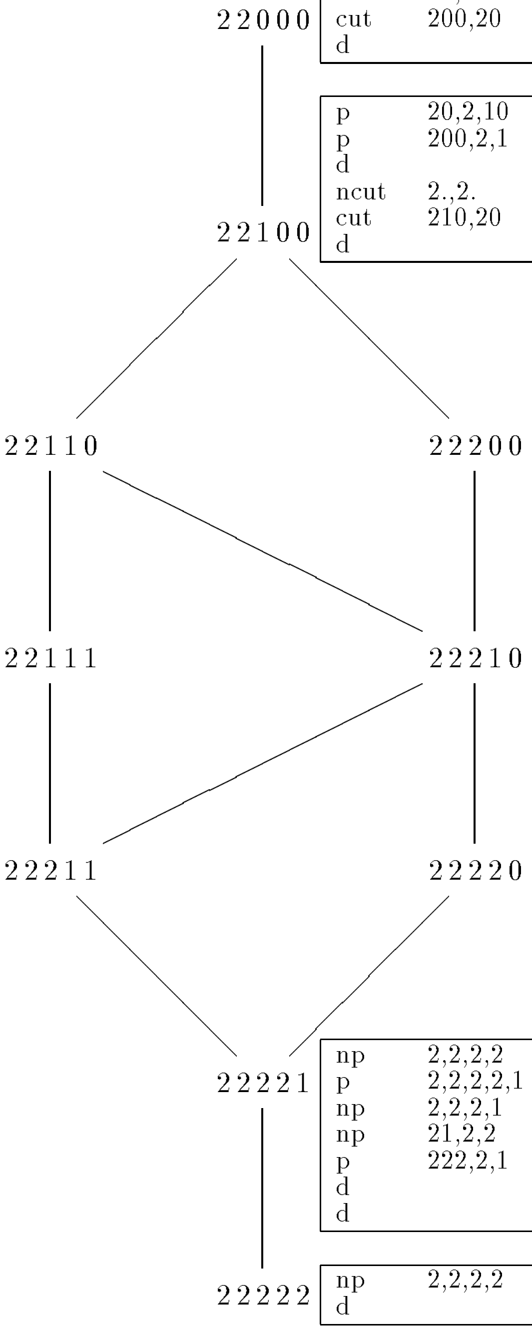


Figure 6.2: Inequalities for 2ECON$(K_5; r)$



2 2 0 0 0

ncut	20,20	1	6
cut	200,20	2	6
d		2	2

2 2 1 0 0

p	20,2,10	3	6
p	200,2,1	3	2
d		1	1
ncut	2.,2.	1	6
cut	210,20	2	4
d		2	2

2 2 1 1 0

p	2,2,1,10	4	2
p	20,2,1,1	4	2
p	210,2,1	3	4
d		1	2
ncut	2.,2.	1	6
cut	211,20	2	2
d		2	2

2 2 2 0 0

np	20,2,2	2	6
ncut	20,20	1	6
cut	220,20	2	6
d		2	3

2 2 1 1 1

p	2,2,1,1,1	5	1
p	211,2,1	3	6
d		1	3
ncut	21,21	1	6
d		2	2

2 2 2 1 0

"rc+1"	2,2,2,1	8	1
np	2,2,2,1	3	1
p	2,2,2,10	4	1
p	20,2,2,1	4	3
np	2.,2,2	2	6
p	220,2,1	3	3
d		1	1
ncut	21,20	1	6
cut	221,20	2	3
d		2	3

2 2 2 1 1

p	2,2,2,1,1	5	1
np	2,2,2,1	3	2
np	21,2,2	2	6
p	221,2,1	3	6
d		1	2
ncut	21,21	1	6
d		2	3

2 2 2 2 0

np	2,2,2,2	3	1
np	20,2,2	2	12
cut	222,20	2	4
d		2	4

2 2 2 2 1

np	2,2,2,2	3	1
p	2,2,2,2,1	5	1
np	2,2,2,1	3	4
np	21,2,2	2	12
p	222,2,1	3	4
d		1	1
d		2	4

2 2 2 2 2

np	2,2,2,2	3	5
d		2	5

Figure 6.3: 2NCON($K_5; r$) polyhedra

7 Computational Results

For applied mathematicians, the ultimate test of the power of a theory is its success in helping solve the practical problems for which it was developed. In our case, this means that we have to determine whether the polyhedral theory for the survivable network design problem can be used in the framework of a cutting plane algorithm to solve the design problems of the sizes arising in practice. The results reported in Grötschel, Monma and Stoer (1992b) show that the design problems for the LATA networks arising at Bellcore can be solved to optimality quite easily. There is good reason to hope that other network design problems of this type can also be attacked successfully with this approach. Moreover, the known heuristics also seem to work quite well, at least for the low connectivity case.

7.1 Outline of the Cutting Plane Algorithm

We have already mentioned in Section 6.3 how a cutting plane approach for our problem works. Let us repeat this process here a little more formally.

We assume that a graph $G = (V, E)$ with edge cost $c_e \in \mathbf{R}$ for all $e \in E$ and node types $r_v \in \mathbf{Z}_+$ for all $v \in V$ is given. Let $k := \max\{r_v \mid v \in V\}$. We want to solve either the kNCON or the kECON problem for G and r and the given cost function, i.e., we want to solve

$$\begin{array}{ccc} \min c^T x & \text{or} & \min c^T x \\ x \in k\text{NCON}(G; r) & & x \in k\text{ECON}(G; r). \end{array}$$

We do this by solving a sequence of linear programming relaxations that are based on the results we have described in Section 7. The initial LP (in both cases) consists of the degree constraints and trivial inequalities, i.e.,

$$\text{(7.1)} \qquad \begin{array}{ll} \min c^T x & \\ x(\delta(v)) \geq r_v & \text{for all } v \in V; \\ 0 \leq x_e \leq 1 & \text{for all } e \in E. \end{array}$$

If a current LP-relaxation has been solved and z is a basic optimum solution, we check whether z is in $k\text{NCON}(G; r)$ or $k\text{ECON}(G; r)$. If it is, the problem is already solved to optimality. Otherwise we call our separation routines and try to find inequalities in one of the classes described in Section 6 that are violated by z. If one of the separation algorithms is successful, we add the inequalities found to the current LP and repeat the procedure. If no violated inequality can be identified, there are two options. We may simply stop and report a lower bound or we may resort to an enumerative procedure like branch and bound.

The best option to select in the case that the cutting plane phase has not produced an optimum solution of the kECON or kNCON problem depends on the demands of practice. Before starting the cutting plane algorithm, one usually runs some heuristics to produce (hopefully) good feasible solutions, see Section 4 for a description of such heuristics. By comparing the lower bound L from the cutting plane algorithm with the upper bound U from the heuristics, one can easily get an idea about the quality of the solutions. If the percentage deviation of these values, usually taken as $100 * (U - L)/L$, is less than some threshold, say 5%, and if the cost data were somewhat fuzzy anyway, the best heuristic solution might simply be considered as an appropriate good solution of the practical problem. If the data are precise and an optimum solution is needed, branch and bound has a good chance to terminate in a "reasonable" amount of time.

If, however, the deviation of U and L is large, no simple advice can be given. Either the heuristic or the cutting plane algorithm or both may have produced poor bounds. Further research is usually necessary to determine the reasons for failure, to detect special structures of the problem in question that may have resulted in traps for the heuristics or poor performance of the cutting plane algorithms. If such structural insight can be obtained, one has a good chance to improve the result, at least for the current problem instance.

The last case is definitely unsatisfactory, but since we are dealing with hard problems, we have to expect such behavior every now and then. For our real world applications we can report that, for the LATA network design problems, the cutting plane algorithm always found an optimum integral solution, except in three cases that were easily solved by branch and bound or manual interaction. Random problems with low and high connectivity requirements and random cost structure were solved to optimality extremely quickly. But there is one large scale practical problem with 494 nodes, and 1096 edges and highly structured topology and costs, where we ran into considerable difficulties. Using special purpose separation heuristics etc., we were finally able to solve two versions of the problem to optimality, with quite some effort however.

7.2 Implementation Details

We will not discuss implementation details of the heuristics outlined in Section 4; see Monma and Shallcross (1989) for details. This is quite straightforward, although of course, the use of good data structures and search strategies may result in considerable running time improvements.

We will focus here on implementation issues of the cutting plane algorithm, a number of which are vital for obtaining satisfactory running time performances. Before starting the cutting plane algorithm we try to reduce the size of the problem instance by decomposing it. In fact, the practical problems we solved have rather sparse graphs of possible direct links, and the survivability requirements often force certain edges to be present in every feasible solution. Such edges can be fixed and removed from the problem by appropriately changing certain node types. This removal may break the original problem into several smaller ones that can be solved independently.

There are further ways of decomposing a problem into independent subproblems like decomposing on articulation nodes, on cut sets of size two, and on articulation sets of size two. In each of these cases one can perform the decomposition or determine that no such decomposition is possible, using polynomial time methods like depth-first search or connectivity algorithms. All of this is quite easy, though a precise description of the necessary transformations would require considerable space. Details can be found in Grötschel, Monma and Stoer (1992b), and Stoer (1992).

The main purpose of this decomposition step is to speed up the computation by getting rid of some trivial special cases that the cutting plane algorithm does not need to check any more, and by reducing the sizes of the problems to be solved. At the end of this preprocessing phase we have decomposed the original problem into a list of subproblems for which we call the cutting plane algorithm. The optimal solution of the original problem can then be composed from the optimal solutions of the subproblems in a straightforward manner.

An issue of particular importance is the implementation of the separation algorithms. Good heuristics have to be found that "solve" the separation problems for those classes that are not known to be separable in polynomial time. And even if polynomial exact separation routines are known, separation heuristics may help considerably to speed up the overall program. Further problems are to determine the order in which the heuristic and exact separation routines are to be called, when to stop the separation process in case many violated inequalities have been found, which cutting planes to add and which to eliminate from the current LP. These issues can not be decided by "theoretical insight" only. Practical experience on many examples is necessary to come up with recipes that result in satisfactory overall performance of such an algorithm. We outline some of the techniques used in the sequel.

Let G be a graph with node types $r \in \mathbf{Z}_+^V$, and let y be some point in $\mathbf{R}^E$ with $0 \leq y_e \leq 1$. Our aim is to find a partition (resp. node partition, lifted r-cover) inequality violated by this point. By the NP-completeness results mentioned in Section 6.3, it seems hopeless to find an efficient exact algorithm for the separation of these inequalities; therefore we have to use heuristics. Nevertheless, it is possible to solve the separation problem for a certain subclass of these inequalities, namely cut constraints (resp., node cut and r-cover constraints) in polynomial time. So in our heuristics we often use "almost violated" inequalities of these subclasses and transform them into violated inequalities of the larger class. Here an almost violated inequality is an inequality $a^T x \geq b$ with $a^T y \leq b + \alpha$ for some "small" parameter α (we used $\alpha = 0.5$). $a^T x \geq b$ is a violated inequality, if $a^T y < b$.

The heuristic that we applied has the following general form:

(7.2) Heuristic for Finding Violated Partition Inequalities

1. *Shrink all or some edges $e \in E$ with the property that any violated partition inequality using this edge can be transformed into some at-least-as-violated partition inequality not using this edge. ("Using e" means; e has coefficient 1 in the partition inequality.)*

2. *Find some violated or almost violated cut constraints in the resulting graph.*

3. *Attempt to modify these cut constraints into violated partition inequalities.*

Exactly the same approach is used for separating node partition (6.5) and lifted r-cover inequalities (6.9), except that we have to use other shrinking criteria and, in Step 2, plug in the appropriate subroutine for separating node cut (6.2), resp., r-cover constraints (6.8)(i).

Shrinking is important for reducing graph sizes before applying the Gomory-Hu algorithm for finding cut constraints, which has the rather high complexity of $O(|V|^4)$. If we are looking for related partition inequalities, we test whether edge $e = uv$ satisfies one of the following shrinking criteria.

(7.3) Shrinking Criteria

1. $y_e \geq q := max\{r_w : w \in V\}$

2. $y_e \geq r_v$ and $y_e \geq y(\delta(v)) - y_e$

3. $y_e \geq max\{y(\delta(v)) - y_e, y(\delta(u)) - y_e\}$ *and there is a node* $w \notin \{u, v\}$ *with* $r_w \geq max\{r_u, r_v\}$.

If these criteria are satisfied for edge e, we shrink it by identifying u and v, giving type $r(\{u, v\})$ to the new node, and identifying parallel edges by adding their y-values.

It can be shown, that if cases (7.3)1. or 2. apply, then any violated partition inequality using e can be transformed into some at-least-as-violated partition inequality not using e. In case

(7.3)3., edge e has the same property with respect to cut inequalities. Similar shrinking criteria can be found for node partition inequalities and lifted r-cover inequalities.

In the reduced graph G' we now find violated or almost violated cut constraints (resp., node partition and r-cover constraints) using the Gomory-Hu algorithm (or, for r-cover constraints the Padberg-Rao algorithm). These inequalities, defined for G', are transformed back into the original graph G. For instance, a cut inequality in G', $x(\delta_{G'}(W')) \geq r(W')$ is first transformed into a cut inequality $x(\delta_G(W)) \geq r(W)$ in G by blowing up all shrunk nodes in W'. This provides the enlarged node set W. Secondly, this cut inequality is transformed into a (hopefully) violated partition inequality by splitting W or $V \setminus W$ into smaller disjoint node sets $W_1, \ldots, W_p$.

We also check whether the given cut inequality satisfies some simple necessary criteria for defining a facet of kECON$(G;r)$ (or kNCON$(G;r)$). If this is not so, it can usually be transformed into a partition inequality that defines a higher-dimensional face of the respective polyhedron. A similar approach is taken for node partition and lifted r-cover inequalities. More details can be found in Grötschel, Monma and Stoer (1992b) and in Stoer (1992). Typically, in the first few iterations of the cutting plane algorithm, the fractional solution y consists of several disconnected components. So, y violates many cut and partition inequalities but usually no lifted r-cover inequalities. We start to separate lifted r-cover inequalities only after the number of violated partition inequalities found drops below a certain threshold. Node partition inequalities are used only after all other separation algorithms failed in finding more than a certain number of inequalities.

To keep the number of LP constraints small, all inequalities in the current LP with non-zero slack are eliminated. But since all inequalities ever found by the separation algorithms are stored in some external pool they can be added again if violated at some later point.

7.3 Computational Results for Low-Connectivity Problems

In this section we describe the computational results based on the practical heuristics described in Section 4 and the cutting plane approach described earlier in this section. We consider the low-connectivity case with node types in $\{0, 1, 2\}$ here and the high connectivity case in the next section. All running times reported are for a SUN 4/50 IPX workstation (a 28.5 MIPS machine). The LP-solver used is a research version of the CPLEX-code provided to us by Bixby (1991). This is a very fast implementation of the simplex algorithm.

To test our code, network designers at Bellcore provided the data (nodes, possible direct links, costs for establishing a link) of seven real LATA networks that were considered typical for this type of application. The sizes ranged from 36 nodes and 65 edges to 116 nodes and 173 edges; see Table 7.1. The problem instances LATADL, LATADS, and LATADSF are defined on the same graph. The edges have the same costs in each case, but the node types vary. Moreover, in LATADSF, 40 edges were required to be in the solution. (The purpose was to check how much the cost would increase if these edges had to be used, a typical situation in practice, where alternative solutions are investigated by requiring the use of certain direct links.)

Table 7.1 provides information about the problems. Column 1 contains the problem names. For the original graphs, columns 2, 3, and 4 contain the numbers of nodes of type 0, 1, and 2, respectively; column 5 lists the total number of nodes, column 6 the number of edges and the number of edges required to be in any solution (the forced edges). All graphs were analysed by our preprocessing procedures described in Section 7.2. Preprocessing was very successful. In fact, in every case, the decomposition and fixing techniques ended up with a single, much smaller graph obtained from the original graph by splitting off side branches consisting of nodes of type 1, replacing paths where all interior nodes are of degree 2, by a single edge, etc. The data of the resulting reduced graphs are listed in columns $6, \ldots, 10$ of Table 7.1.

	Original Graphs					**Reduced Graphs**				
Problem	**0**	**1**	**2**	**Nodes**	**Edges**	**0**	**1**	**2**	**Nodes**	**Edges**
LATADMA	0	12	24	36	65/0	0	6	15	21	46/4
LATA1	8	65	14	77	112/0	0	10	14	24	48/2
LATA5S	0	31	8	39	71/0	0	15	8	23	50/0
LATA5L	0	36	10	46	98/0	0	20	9	29	77/1
LATADSF	0	108	8	116	173/40	0	28	11	39	86/26
LATADS	0	108	8	116	173/0	0	28	11	39	86/3
LATADL	0	84	32	116	173/0	0	11	28	39	86/6

Table 7.1 Data for LATA Problems

To give a visual impression of the problem topologies and the reductions achieved, we show in Figure 7.1 a picture of the original graph of the LATADL problem (with 32 nodes of type 2 and 84 nodes of type 1) and in Figure 7.2 a picture of the reduced graph (with 39 nodes and 86 edges) after preprocessing. The nodes of type 2 are displayed by squares, and the nodes of type 1 are displayed by circles. The 6 forced edges that have to be in any feasible solution are drawn bold.

LATA1 is a 2ECON problem, while the other six instances are 2NCON problems. All optimum solutions of the 2ECON versions turned out to satisfy all node-survivability constraints and thus were optimum solutions of the original 2NCON problems – with one exception. In LATA5L one node is especially attractive because many edges with low cost lead to it. This node is an articulation node of the optimum 2ECON solution. In the following, LATA5LE is the 2ECON version of problem LATA5L.

Table 7.2 contains some data about the performance of our code on the eight test instances. The entries from left to right are:

IT	number of iterations (= calls to the LP-solver)
P	number of partition inequalities (6.4) used in the cutting plane phase
NP	number of node partition inequalities (6.5) used in the cutting plane phase
RC	number of lifted r-cover inequalities (6.9) used in the cutting plane phase
C	value of the optimum solution after termination of the cutting plane phase
COPT	optimum value
GAP	$100 \times (\text{COPT} - \text{C})/\text{COPT}$ (= percent relative error at the end of the cutting plane phase)
T	total running time including input, output, preprocessing, etc., of the cutting plane phase (not including branch & cut), in rounded
BN	number of branch & cut nodes generated
BD	maximum depth of the branch & cut tree
BT	total running time of the branch & cut algorithm including the cutting plane phase, in seconds

Problem	**IT**	**P**	**NP**	**RC**	**C**	**COPT**	**GAP**	**T**	**BN**	**BD**	**BT**
LATADMA	12	65	3	7	1489	1489	0	1			
LATA1	4	73	0	1	4296	4296	0	1			
LATA5S	4	76	0	0	4739	4739	0	1			
LATA5LE	7	120	0	0	4574	4574	0	1			
LATA5L	19	155	12	0	4679	4726	0.99	2	4	2	4
LATADSF	7	43	0	0	7647	7647	0	1			
LATADS	17	250	0	4	7303.60	7320	0.22	4	28	9	17
LATADL	14	182	0	28	7385.25	7400	0.20	3	32	10	21

Table 7.2. Performance of branch & cut on LATA problems

We think that it is worth noting that each of these real problems, typical in size and structure, can be solved on a 28-MIPS machine in less than thirty seconds including all input and output routines, drawing the solution graph, branch and cut, etc.

A detailed analysis of the running times of the cutting plane phase is given in Table 7.3. All times reported are in percent of the total running time TIME (without the branch & cut phase). The entries from left to right are:

PT	time spent in the preprocessing phase (in percent)
CT	time spent in the separation routines (in percent)
LPT	time used by the LP-solver (in percent)
MT	miscellaneous time for input, output, drawing, etc. (in percent)
TT	total time (in seconds)
TT\RED	total time (in seconds) of the algorithm when applied to the original instance without prior reduction by preprocessing

The last column TT\RED shows the running times of the cutting plane phase of our algorithm applied to the full instances on the original graphs (without reduction by preprocessing). By comparing the last two columns, one can clearly see that substantial running time reductions can be achieved by our preprocessing algorithms on the larger problems.

PROBLEM	**PT%**	**LPT%**	**CT%**	**MT%**	**TT**	**TT\RED**
LATADMA	2.0	39.2	41.2	17.6	1	1
LATA1	3.8	34.6	34.6	26.9	1	4
LATA5S	3.8	34.6	34.6	26.9	1	1
LATA5LE	0.0	42.9	41.1	16.1	1	1
LATA5L	0.7	37.1	55.2	7.0	2	5
LATADSF	2.1	21.3	57.4	19.2	1	4
LATADS	0.0	44.7	49.0	6.4	4	17
LATADL	1.0	26.3	66.2	6.5	3	18

Table 7.3. Running times of cutting plane algorithm on LATA problems

A structural analysis of the optimum solutions produced by our code revealed that – except for LATADSF, LATA5LE, and LATA1 – the optimum survivable networks consist of a long cycle (spanning all nodes of type 2 and some nodes of type 1) and several branches connecting the remaining nodes of type 1 to the cycle. The optimum solution of the LATADL instance in shown in Figure 7.3, with the 2-connected part (the long cycle) drawn bold.

From the view of a telephone network designer, a long cycle connecting all nodes of type 2 is not a desirable feature in a communication network because routing paths are very long, resulting in delays, and because each link has to carry a high traffic load, resulting in high costs for terminal electronics, multiplexers, etc. But since the network installation costs form part of the whole network cost, the lowest network installation cost (as found by our algorithm) provides a lower bound for the whole network cost, and the subgraph minimizing the installation cost could be modified, e.g., by adding some more links of low cost, to produce a network with shorter routing paths between each pair of nodes. This is the design approach taken in the software package distributed by Bellcore (1988); first a survivable network topology of low cost in computed by heuristics, then this topology is modified to account also for costs associated with the expected traffic in the network.

We ran a few tests on randomly generated problems of higher density and 50-100 nodes. Here our code did perform reasonably well but not as well as on sparse problems. (That is not of great importance, since our goal was to solve real-world problems and not random problems.) More serious is a dramatic increase in running time when many nodes of type 0 are added, as is the case in the ship problem treated in the next section. Here our cutting plane code for the low-connectivity case takes very long before the intermediate fractional solutions become connected. For such cases new separation heuristics have to be developed that perform a more sophisticated structural analysis of the given instance. But the problems that we address here mainly and that come up in the design of fiber optic telephone networks have very few nodes of type 0, if any.

Another motivation for our work was to find out how well the heuristics of Monma and Shallcross (1989) described in Section 4 perform. It turned out that they do very well. Table 7.4 compares the values CHEUR of the solutions produced by the heuristics with the optimum values COPT computed by our code. The percent relative error GAP (= 100 $\times$ (CHEUR - COPT)/COPT) is always below 1.5%. In three cases the heuristics found an optimum solution. This result definitely justifies the present use of these heuristics in practice. We note that these heuristics are very fast, typically taking only a few seconds on a IBM PC/AT.

PROBLEM	**COPT**	**CHEUR**	**GAP**
LATADMA	1489	1494	0.34
LATA1	4296	4296	0
LATA5S	4739	4739	0
LATA5LE	4574	4574	0
LATA5L	4726	4794	1.44
LATADSF	7647	7727	1.05
LATADS	7320	7361	0.56
LATADL	7400	7460	0.81

Table 7.4: Comparison of heuristic values with optimal values

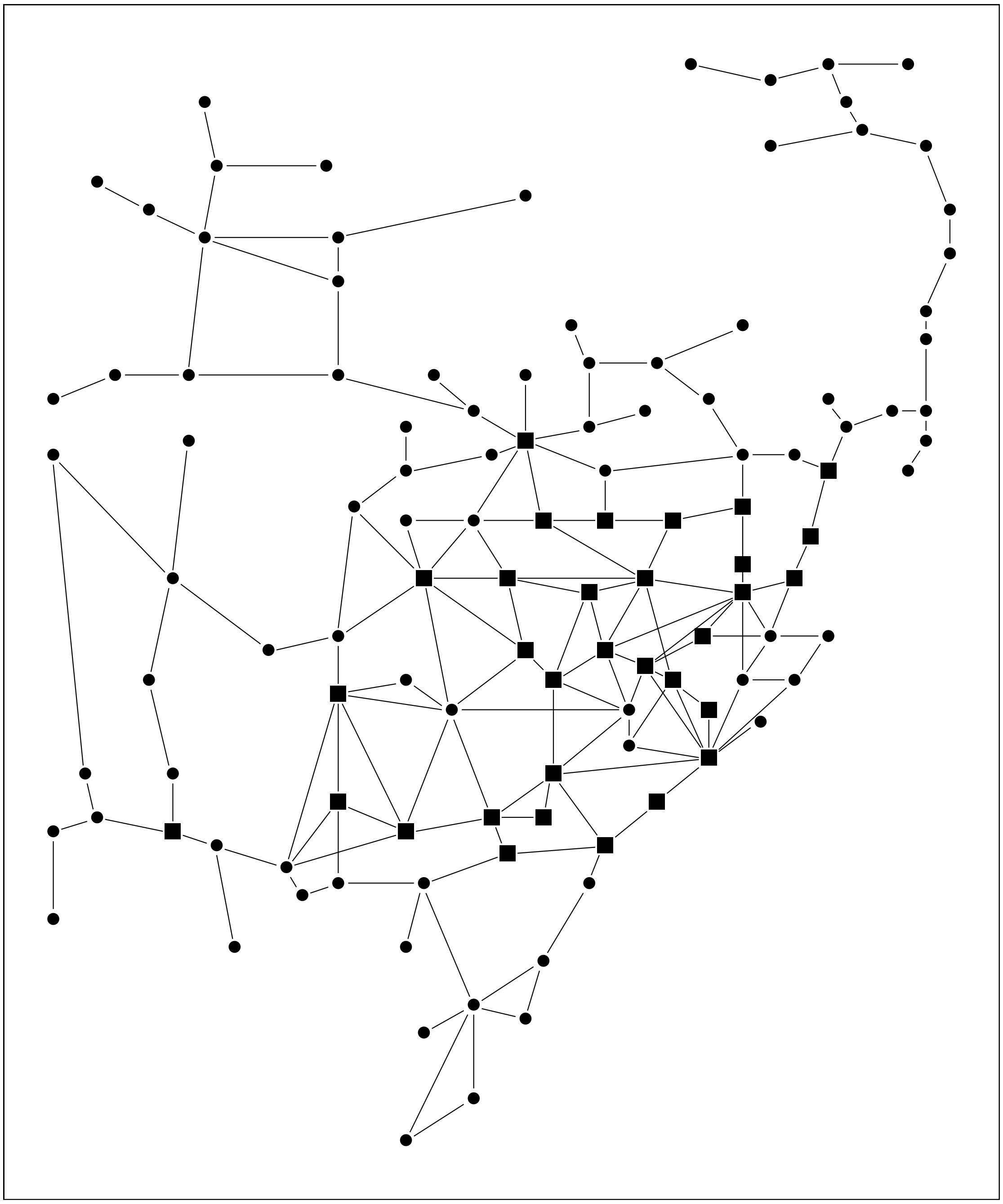


Figure 7.1: Original graph of LATADL-problem

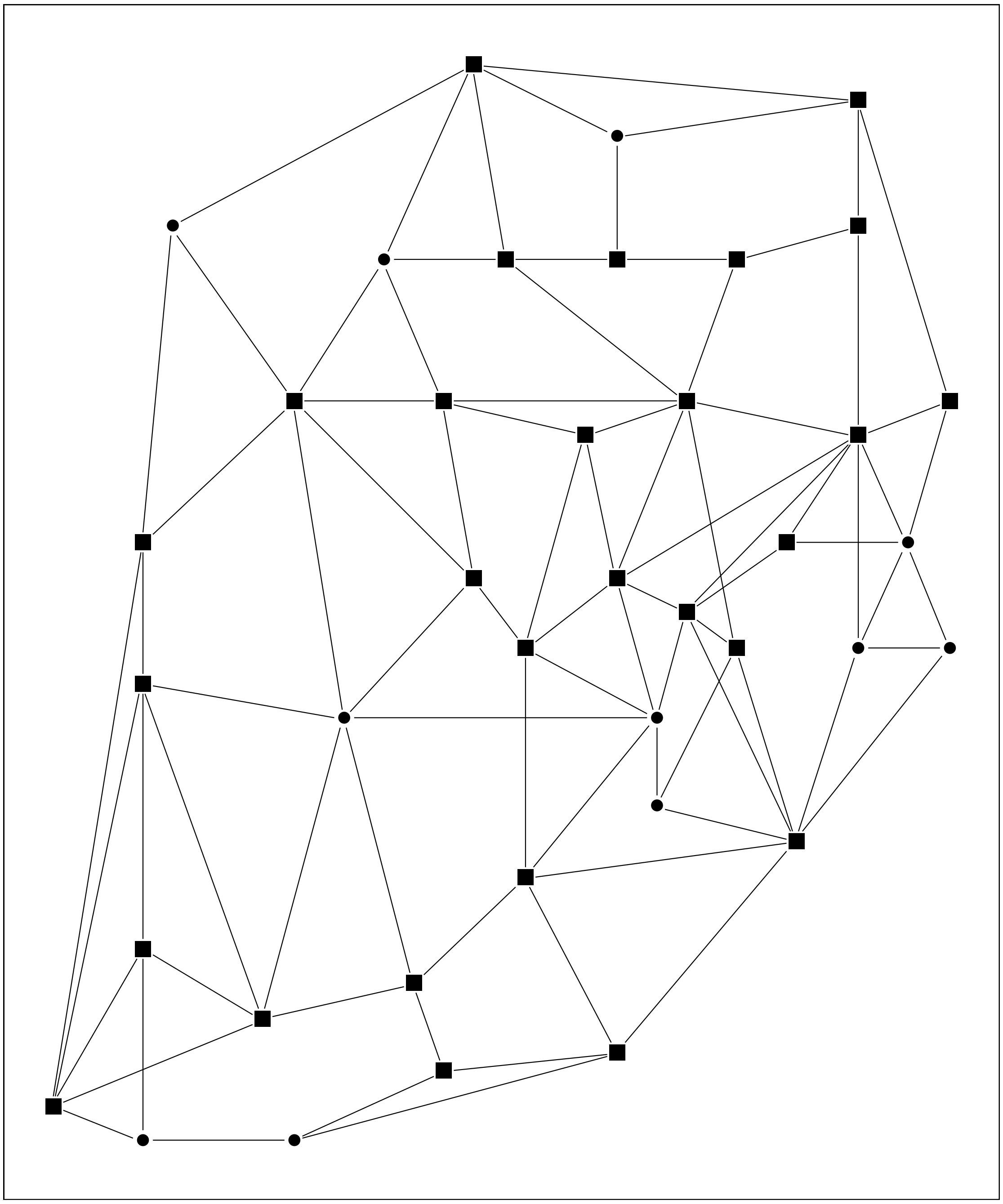


Figure 7.2: Reduced graph of LATADL-problem

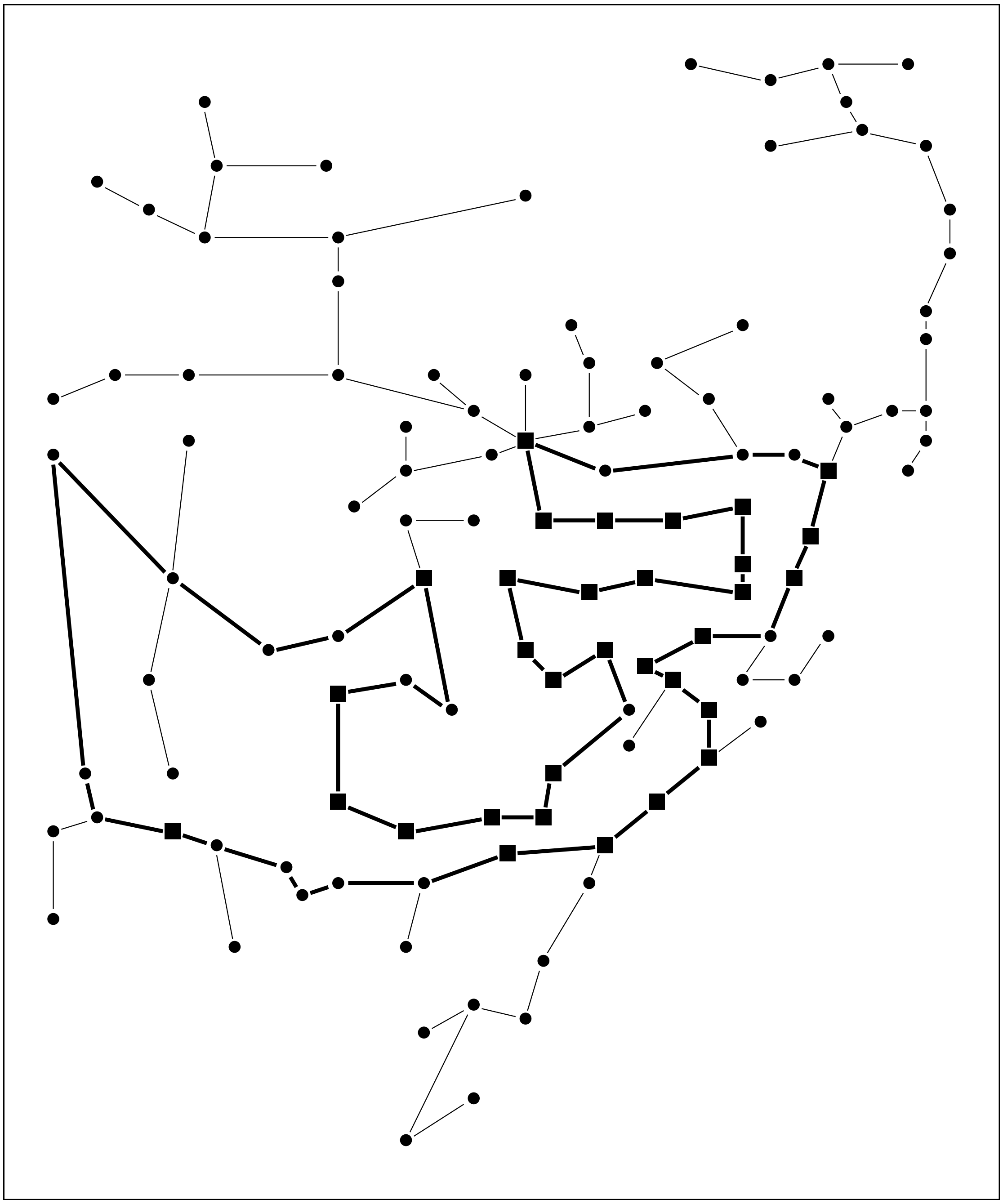


Figure 7.3: Solution of LATADL-problem

7.4 Computational Results for High-Connectivity Problems

At present, we have a first preliminary version of a code for solving survivability problems with higher connectivity requirements. In order to test our code for general kNCON problems, we first used a set of random problems. Later, we also obtained test data for a real-world 3NCON problem, which arose in the design of a communication network on a ship. Both types of test problems have their "drawbacks", however. The random problems turned out to be too easy (most of them were already solved in the first iteration), and the ship problem confronted us with so many new difficulties (with respect to space, running time, and quality of solutions) that we have to redesign our separation strategies completely to solve variants of the ship problem to optimality.

7.4.1 Random Problems

We first report about our computational results on random kECON problems. We used the same set of random data as Ko and Monma (1989) used for their high-connectivity heuristics. So we will be able to compare results later.

The test set of Ko & Monma consists of five complete graphs of 40 nodes and five complete graphs of 20 nodes, whose edge costs are independently drawn from a uniform distribution of real numbers between 0 and 20. For each of these 10 graphs, a minimum-cost k-edge connected subgraph for $k = 3, 4, 5$ is to be found. The next table reports the number of iterations (minimum and maximum) and the average time taken by our code to solve these problems for $k = 3, 4,$ and 5, respectively. Only the time for the cutting plane phase is given.

# Nodes	**# Iterations**			**Average Time (secs)**		
$k =$	3	4	5	3	4	5
20 nodes:	1-2	1-5	1-4	0.43	0.51	0.58
40 nodes:	1-2	1-2	1-4	1.54	1.95	2.36

All problems except one 3ECON instance on 20 nodes were solved in the cutting plane phase. In fact 20 of the 30 problems were already solved in the first iteration with the initial LP (7.1). For the instances not solved in the first iteration, at most four lifted r-cover inequalities (6.9) had to be added to obtain the optimal solution. Except for one 3ECON instance, no partition inequalities were added. So, the average solution time is mainly the solution time for the first LP.

All optimal solutions for the kECON problems were at the same time feasible and hence optimal for the corresponding kNCON problems, except the one 3ECON problem which could not be solved in the cutting plane phase. There the optimal solution (obtained by branch & cut) is 3-edge connected, but not 3-node connected.

These excellent results were surprising, because we always thought high-connectivity problems to be harder than low-connectivity problems. But this does not seem to be true for random costs.

The high-connectivity heuristics of Ko and Monma did not perform quite as well as the low-connectivity heuristics, but still reasonably well. The relative gap between the heuristic (h) and the optimal solution value (o), namely $100 \times (h - o)/o$, computed for the above set of random problems, ranged between 0.8 and 12.8 with an average of 6.5 % error (taken over all problems).

7.4.2 Ship Problems

One real-world application of survivable network design, where connectivities higher than two are needed, is the design of a fiber communication network that connects locations on a military ship containing various communication systems. The reason for demanding high survivability of this network is obvious.

The problem of finding a high-connected network topology minimizing the cable installation cost can be formulated as a 3NCON problem. We will describe the characteristics of this problem in the following.

We obtained the graph and edge cost data of a generic ship model. It has the following features. The graph of possible link installations has the form of a three-dimensional grid with 15 layers, 494 nodes, and 1096 edges, which is depicted in Figure 7.4. The problem to be solved in this graph is a 3NCON problem with the following node types and costs.

Of the grid's 494 nodes, only 33 are nonzero type, called "special nodes". They are drawn by filled circles or triangles. The 33 special nodes symbolize the various communication systems to be interconnected by the network. To evaluate the dependence of network topology cost on the required survivability, the ship problem appears in three different versions depending on the node types of the 33 special nodes. The three nodes depicted by triangles in the tower of the ship always have type 3, the other 29 special nodes are all given either type 1, type 2, or type 3. We call the three resulting versions of the ship problem "ship13", "ship23", and "ship33", respectively. The remaining 461 nodes are nodes of type 0. They represent possible fiber junction boxes where the fiber cable may be routed.

The cost structure is highly regular. The costs are proportional to the distances between nodes, with the feature that horizontal distances are much higher than vertical distances.

(The grid shown in Figure 7.4 has been scaled. Also, contrary to the graphical representation, the horizontal layers do not always have the same distance from each other.) With this cost structure, it is much cheaper to route vertically than horizontally. Since there exist many shortest paths between any two nodes, there will also exist many optimum solutions to the survivable network problem. So the problem is highly degenerate. Degeneracy together with the size of the ship problem caused us to run into difficulties. In fact, when we first applied our code to the "ship13" problem, with the initial LP consisting only of the degree constraints for the special nodes, the fractional solutions did not get connected for a long time.

Our first idea was to heuristically reduce the size of the problem in some way. Unfortunately, none of the decomposition techniques described earlier applied, except at the tower of the ship, where nodes of type 3 are separated by a cut of size 3. We cut out some of the "unnecessary" nodes of type 0 in the lower left and right hand corner of the grid, and also deleted some of the horizontal layers of the grid containing only nodes of type 0.

It is not obvious at all that corners of a grid may be cut out and layers may be deleted without affecting the optimum objective function value of the problem. We could prove such a result only for Steiner tree problems (1NCON problems), not for 2NCON or 3NCON problems. But nevertheless, we used these reductions heuristically to cut down problem sizes in the hope that some optimal solution of the original graph is still contained in the reduced graph. For the ship23 problem, the optimal solution of the reduced problem turned out to be optimal for the nonreduced problem, too.

Figure 7.5 shows the reduced graph of the "ship13" problem. The result of the reductions can be seen from Table 7.5, whose columns list, from left to right, the problem names, and, for the original ship graph and the reduced ship graphs, the number of nodes of type 0, 1, 2, and 3, the total number of nodes and the total number of edges/number of forced edges. The forced edges are those edges contained in some cut of size 3 separating two nodes of type 3, which must be contained in any feasible solution.

An optimal solution for the reduced "ship23" problem is shown in Figure 7.6.

	Original Graph					**Reduced Graph**				
Problem	**0**	**1**	**2**	**3**	**Edges**	**0**	**1**	**2**	**3**	**Edges**
ship13	461	30	0	3	1096/0	128	28	0	3	325/3
ship23	461	0	30	3	1096/0	249	0	30	3	607/3
ship33	461	0	0	33	1096/0	300	0	0	33	719/9

Table 7.5: Sizes of ship problems

Table 7.5 shows that the reductions are enormous, yet there are still many more nodes of type 0 than nodes of nonzero type in each problem.

When we applied our code to the reduced graphs, the fractional solutions still looked frequently like paths beginning at some special node and ending in some node of type 0. To cure this problem, we made use of the following type of inequalities.

$$x(\delta(v)\setminus\{e\}) \geq x_e$$

for all nodes v of type 0 and all $e \in \delta(v)$. These inequalities (we call them **con0 inequalities)** describe algebracially that nodes of type 0 do not have degree 1 in an edge-minimal solution. This is not true for all survivable networks, but it is true for the optimum solution if all costs are positive. So, although these inequalities are not valid for the kNCON polytope, we used them to force the fractional solutions into the creation of longer paths. Another trick that we used to obtain good starting solutions was to use cuts of a certain structure in the start LP.

Table 7.6 gives some preliminary computational results of our cutting plane algorithm on the three reduced and not reduced versions of the ship problem. The entries from left to right are:

PROBLEM	problem name, where "red" means reduced
VAR	number of edges minus number of forced edges
IT	number of LPs solved
PART	number of partition inequalities added
RCOV	number of r-cover inequalities added
LB	lower bound = optimal LP value
UB	upper bound = heuristic value
GAP	(ub-lb)/lb in percent
TIME	in minutes:seconds

PROBLEM	**VAR**	**IT**	**PART**	**RCOV**	**LB**	**UB**	**GAP**	**TIME**
ship13	1088	3252	777261	0	211957.1	217428	2.58	10122:35
ship23	1088	15	4090	0	286274	286274	0	27:20
ship33	1082	42	10718	1	461590.6	483052	4.64	55:26
ship13red	322	775	200570	0	217428	217428	0	426:47
ship23red	604	12	2372	0	286274	286274	0	1:54
ship33red	710	40	9817	0	462099.3	483052	4.53	34:52

Table 7.6: Performance of cutting plane algorithm on ship problems

Although Table 7.6 shows that the code is still rather slow, it could at least solve two of the ship problems. In order to obtain faster results, some more research most be done, especially on finding better starting solutions, devising faster separations heuristics that exploit the problem structure, and, maybe, inventing new classes of inequalities for high-connectivity problems.

Table 7.7 shows the percentage of time spent in the different routines.

PROBLEM	problem name where "red" means reduced
PT	time spent for reduction of problem (in percent)
LPT	time spent for LP solving (in percent)
CT	time spent for separation (in percent)
MT	time on miscellaneous items, input, output, etc (in percent)
TIME	total time in minutes:seconds

PROBLEM	**PT**	**LPT**	**CT**	**MT**	**TIME**
ship13	0.0	75.6	23.9	0.5	10122:35
ship23	0.0	13.1	86.4	0.4	27:20
ship33	0.0	31.2	68.2	0.6	55:26
ship13red	0.0	68.5	30.1	1.4	426:47
ship23red	0.1	39.2	58.6	1.9	1:54
ship33red	0.0	41.1	58.4	0.5	34:52

Table 7.7

We do not understand yet why our code solves the ship23 problem rather easily and why there is still a gap after substantial running time of our cutting plane algorithms for the ship33 problem. Probably, the "small" changes of a few survivability requirements result in more dramatic structural changes of the polyhedra and thus of the inequalities that should be used. It is conceivable that our code has to be tuned according to different survivability requirements settings. We should mention that we did not attempt to solve ship13 and ship33 by entering the branching phase of our code. The gaps are not small enough yet for the enumerative stage to have a decent prospective. Further details of our attempts to solve network design problems with higher connectivity requirements can be found in Grötschel, Monma & Stoer (1992c).

Summarizing our computational results, we can say that for survivability problems with many nodes of type 0 and highly regular cost structure (such as the ship problems) much still remains to be done to speed up our code and enhance the quality of solutions. But for applications in the area of telephone network design, where problem instances typically are of moderate size and contain not too many nodes of type 0, our approach produces very good lower bounds and even optimum solutions in a few minutes. This work is a good basis for the design of a production code for the 2ECON and 2NCON problems coming up in fiber optic network design and a start towards problems with higher and more varying survivability requirements and larger underlying graphs.

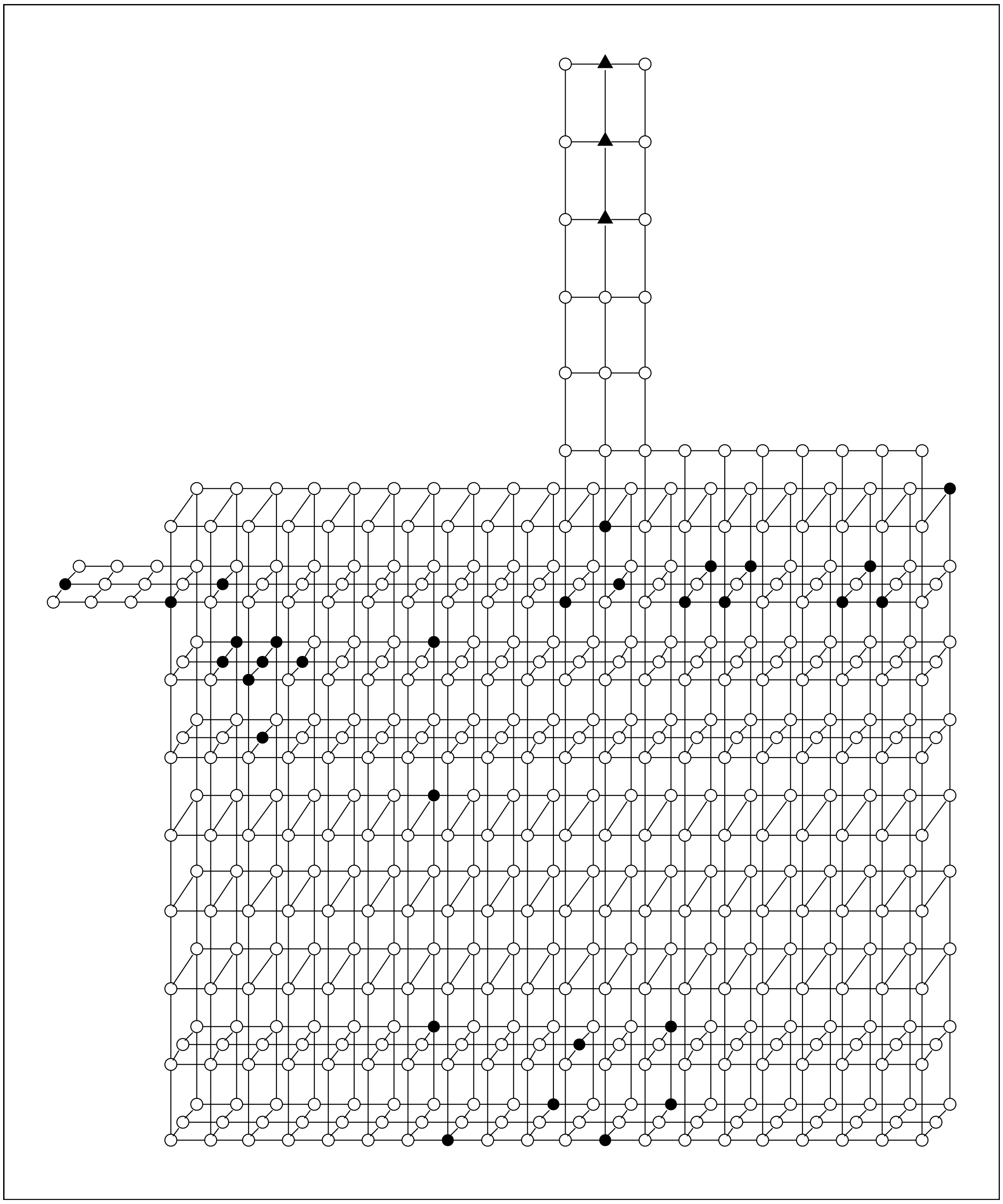


Figure 7.4: Grid graph of the ship problem

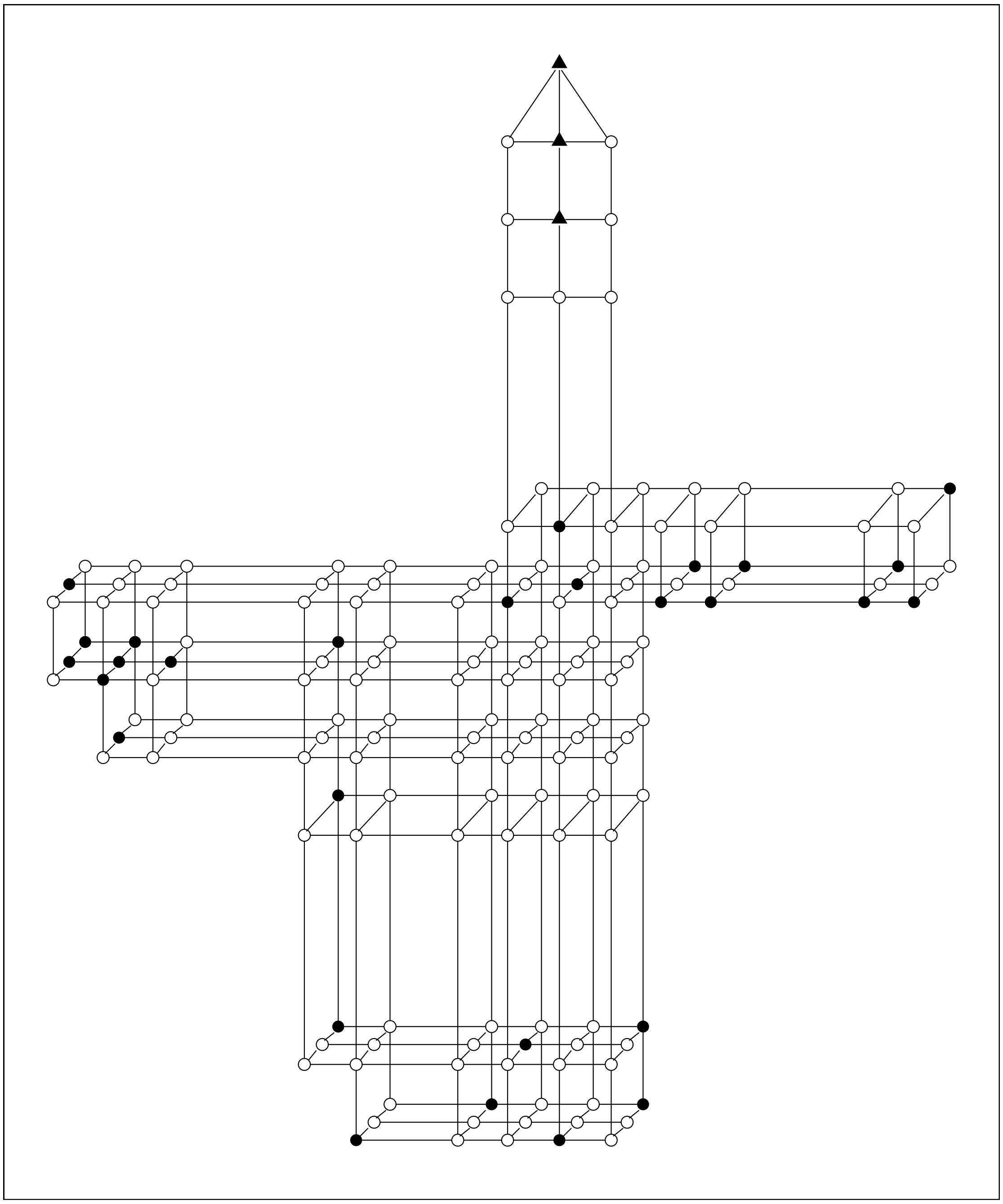


Figure 7.5: Reduced grid graph of the “ship13” problem

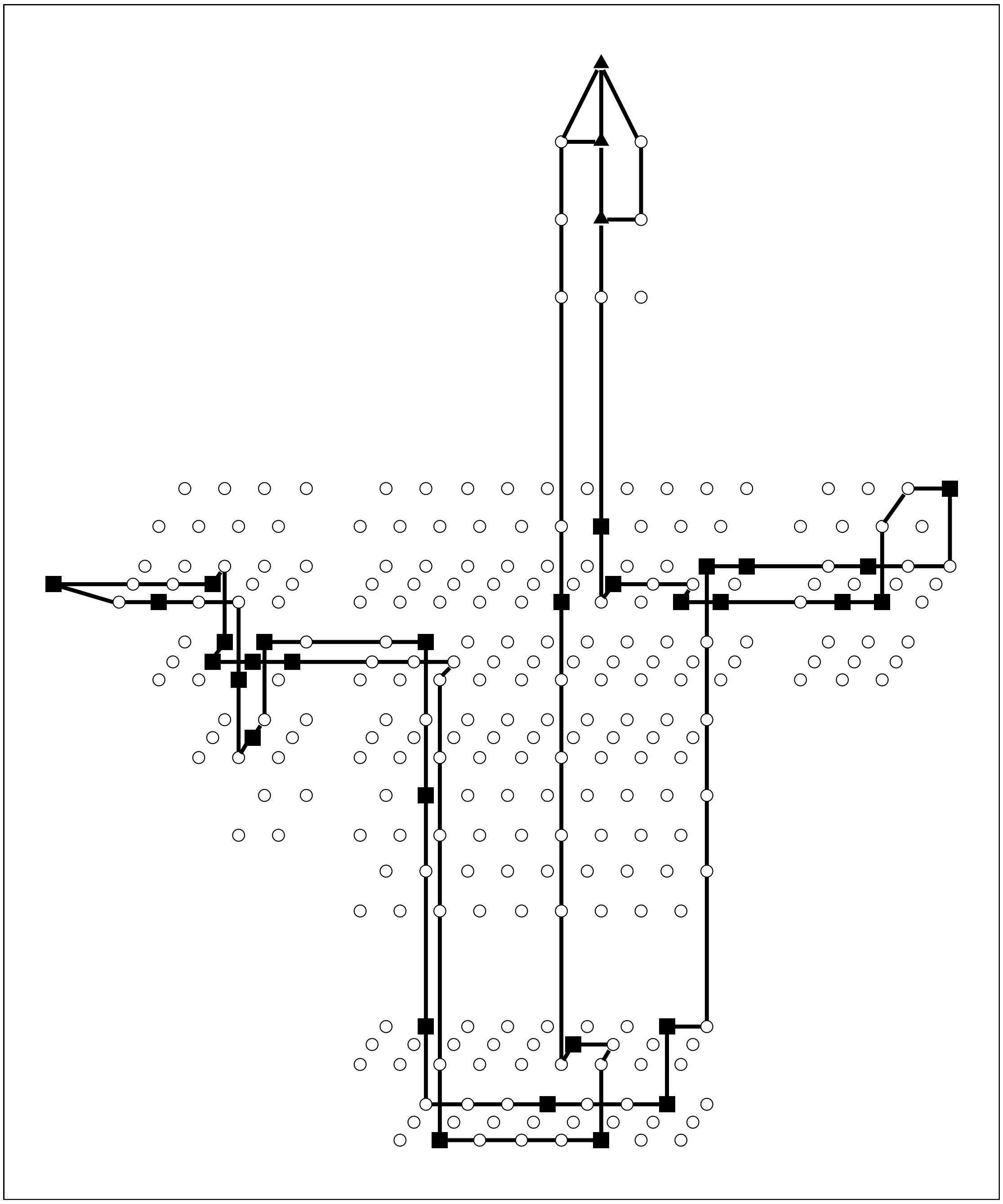


Figure 7.6: Optimum solution of reduced "ship23" problem

8 Directed Variants of the General Model

There are many possible variants of the general model described in Section 3 for the design of networks with connectivity constraints. A natural variant is to consider networks with directed links. As we will see below, there are practical and theoretical reasons for considering survivability in directed graphs.

8.1 Survivability Models for Directed Networks

In order to model directed links, we let $D = (N, A)$ denote a **directed graph** consisting of a set V of **nodes** (just as in the undirected case) and a set A of **directed arcs**. Each **arc** $a = (u, v) \in A$ represents a link directed from node u to node v. For example, this could model certain communications facilities that allow only the one-way transfer of information. Of course, there may be arcs directed each way between any given pair of nodes. Each arc $a \in A$ has a nonnegative **fixed cost** c_a of establishing the link connection. The directed graph may have parallel arcs (in each direction). As before, the cost of establishing a network consisting of a subset $B \subseteq A$ of arcs is the sum of costs of the individual links contained in B. The goal is to build a minimum-cost network so that the required survivability conditions are satisfied.

The survivability requirements demand that the network satisfy the same types of edge and node connectivity requirements as in the undirected case. We simply replace the notion of an undirected path by a directed one. The previous definitions and model formulations are essentially unchanged.

The problem of designing a survivable directed network has not received as much attention in the literature as the undirected case. We briefly summarize some recent efforts along these lines.

Dahl (1991) has given various formulations for the directed survivable network design problem with arc connectivity requirements. He mainly studies bi-Steiner problem which is the problem of finding a minimum-cost directed subgraph that contains two arc-disjoint paths from a given root to each node of a set of terminal nodes. This problem has applications in the design of hierarchical subscriber networks, see Lorentzen & Moseby (1989).

Chopra (1990) found a directed version of the 2ECON problem, which becomes the 2ECON problem after "projection" into a lower-dimensional space. He showed that all partition inequalities and further inequalities can be generated by the projection of certain directed cut inequalities. The idea for the directed model of the kECON problem relies on a theorem of Nash-Williams (1960), that says that all undirected graphs can be oriented in such a way, that the resulting directed graph contains, between each i and j, $\lfloor r_{ij}/2 \rfloor$ arc-disjoint paths, when the underlying undirected graph contains r_{ij} edge-disjoint paths. So Chopra's results can be generalized to higher (even) connectivities.

8.2 Projection

The last remarks show that directed versions of the kECON and kNCON problems are not only interesting in their own right, but they are sometimes also useful in solving their undirected counterparts. We will illustrate this now by pointing out the value of projections. For many combinatorial problems, good polyhedral descriptions can be obtained by transferring the original problem into higher dimensions, that is, by formulating it with additional (auxiliary) variables, which may later be projected away. This was done successfully for the 2-terminal Steiner tree problem in directed graphs, see Ball et al. (1987). There the formulation with auxiliary variables contains a polynomial number of simple constraints, which by projection are turned into an exponential number of "weird" constraints. The general idea of projection was described by Balas and Pulleyblank (1983).

For the 2ECON problem, Chopra (1990) has found a formulation in directed graphs using $2|E|$ integer variables and directed cut constraints, which he called the DECON problem, see (8.1) below. The directed cut constraints (see (8.1)(i)) used in the formulation of the DECON problem have the advantage that they can be separated in polynomial time, whereas the separation of the inequalities appearing in our undirected 2ECON problem is NP-hard.

Projection of the directed cut constraints and nonnegativity constraints of the DECON problem gives a new class of inequalities for the 2ECON problem (we call these Prodon inequalities) which contain as a subclass the partition inequalities (6.4). For the Steiner tree problem (where $r_v \in \{0,1\}$), these new inequalities have been found by Prodon (1985). In the following we show how the Prodon inequalities are derived from the DECON model by projection.

In order to do this, we must first introduce some terminology. Let a graph $G = (V, E)$ and node types $r_v \in \{0, 1, 2\}$ be given, where at least two nodes are of highest (positive) node type. This may either be a 2ECON or a 1ECON problem. From G we construct a directed graph $D = (V, A)$ by replacing each undirected edge ij with two directed edges (i, j) and (j, i). Furthermore, we pick some node $w \in V$ of highest node type. Let $\delta^-(W)$ be the set of arcs directed into node set W. If (x, y) is a solution to the following system of inequalities (where $x \in \mathbf{R}^E$ and $y \in \mathbf{R}^A$),

$$\begin{array}{lllll}
\text{(i)} & y(\delta^-(W)) & \geq 1 & \text{for all } W \subseteq V, \emptyset \neq W \neq V, \text{ with} \\
 & & & \text{con}(W) = 2 \ \text{ (or } r(W) = 1 \text{ and } w \notin W); \\
\text{(ii)} & y_{(i,j)} & \geq 0 & \text{for all } (i,j) \in A; \\
\text{(iii)} & y_{(i,j)} \text{ integral} & & \text{for all } (i,j) \in A; \\
\text{(iv)} & -y_{(i,j)} - y_{(j,i)} + x_{ij} & = 0 & \text{for all } ij \in E; \\
\text{(v)} & x_{ij} & \leq 1 & \text{for all } ij \in E;
\end{array}$$

then the integer vector x is feasible for the 2ECON problem, and vice versa: if some integer vector x is feasible for the 2ECON problem, then an integer vector y can be found so that (x, y) satisfied (8.1)(i)–(v). So the projection of system (8.1) onto x-variables gives a formulation of the 2ECON problem. (Originally, Chopra considered this system without the upper bound constraints.) If no node is of type 2, a feasible vector y is just the incidence vector of a subgraph of D containing a Steiner tree rooted at w. If all nodes are of type 2, then y is the incidence vector of a strongly connected directed subgraph of D ("strongly connected" means that between each distinct pair s, t of nodes there exists a directed (s,t)-path and a directed (t,s)-path).

Without the integrality constraints (iii) and upper bound constraints (v), we obtain a relaxation, which, after projection onto x-variables, gives a relaxation of the 2ECON problem. The projection works as follows. Let us define

- $\mathcal{F}$ as the set of those $W \subseteq V$ that appear in the formulation of inequalities (8.1)(i),
- $b_W \geq 0$ as the variables assigned to each inequality (8.1)(i) for $W \in \mathcal{F}$,
- $a_{ij} \in \mathbf{R}$ as the variables assigned to each equation (8.1)(iv) for $ij \in E$,
- $s(\mathcal{F}; b; i; j)$ as the sum of b_W over all $W \in \mathcal{F}$ with $i \in W$ and $j \notin W$, and
- C as the cone of variables $a \in \mathbf{R}^E$ and $b := (b_W)_{W \in F}$ satisfying

$$\begin{aligned}
a_{ij} &\geq s(\mathcal{F}; b; i; j), \text{ for all } ij \in E \text{ and } W \in \mathcal{F} \\
a_{ij} &\geq s(\mathcal{F}; b; j; i), \text{ for all } ij \in E \text{ and } W \in \mathcal{F}, \\
b &\geq 0.
\end{aligned}$$

If $(a, b) \in C$, and if all inequalities of type (8.1)(i) and all inequalities of type (8.1)(iv) are added with coefficients b_W and a_{ij} respectively, then we obtain an inequality

$$\sum_{(i,j)\in A} u_{(i,j)} y_{(i,j)} + \sum_{ij\in E} a_{ij} x_{ij} \geq \sum_{W\in\mathcal{F}} b_W,$$

where the $u_{(i,j)}$ are non-positive coefficients of the variables $y_{(i,j)}$. In fact, C was defined exactly in such a way, that the $u_{(i,j)}$ are non-positive. The above inequality is valid for the system given by all inequalities (8.1)(i), (i), and (iv). Since $y \geq 0$,

$$\sum_{ij\in E} a_{ij} x_{ij} \geq \sum_{W\in\mathcal{F}} b_W,$$

is valid for $2\mathrm{ECON}(G;r)$. It can also be proved with the general projection technique of Balas and Pulleyblank (1983) that

$$(8.2) \qquad \begin{aligned} \sum_{ij\in E} a_{ij} x_{ij} &\geq \sum_{W\in\mathcal{F}} b_W, \text{ for all } (a,b) \in C \\ x &\geq 0 \end{aligned}$$

is exactly the projection of system (8.1)(i), (ii) and (iv) onto the x-variables. Not all $(a,b) \in C$ are needed in the formulation of (8.2). The following system is clearly sufficient to describe the projection of (8.1) (i), (ii) and (iv) onto x-variables:

$$(8.3) \qquad \begin{array}{lrcll} \text{(i)} & \sum_{ij\in E} a_{ij} x_{ij} & \geq & \sum_{W\in\mathcal{F}} b_W, & \text{for all } b \geq 0 \text{ and} \\ & & & & a_{ij} := \max\{s(\mathcal{F};b;i;j), s(\mathcal{F};b;j;i)\} \\ & & & & \text{for all } ij \in E \\ \text{(ii)} & & x \geq 0. & & \end{array}$$

We call inequalities (8.3)(i) **Prodon inequalities** (induced by b), because this class of inequalities was discovered by Prodon (1985) for $1\mathrm{ECON}(G;r)$.

The class of Prodon inequalities properly contains the class of partition inequalities (7.4). Namely, a partition inequality

$$x[W_1 : \ldots : W_p] \geq \begin{cases} p & \text{if at least two } W_i \text{ contain nodes of type 2} \\ p-1 & \text{otherwise} \end{cases}$$

(where $W_1, \ldots, W_p$ is a partition of V into p node sets with $r(W_i) \geq 1$) can also be written as a Prodon inequality, if b_W is set to 1 for all W_i that are in $\mathcal{F}$ and $b_W := 0$ for all other sets in $\mathcal{F}$. By definition of $\mathcal{F}$, if at least two sets W_i contain nodes of type 2, then $W_i \in \mathcal{F}$ for all W_i, and if only one set, say W_p, contains nodes of type 2 (and therefore the "root" w), then $W_1, \ldots, W_{p-1}$ are in $\mathcal{F}$, but W_p is not. This explains the differing right-hand sides in both cases.

But not every facet-defining Prodon inequality is also a partition inequality. For instance, the inequality depicted in Figure 8.1 is not a partition inequality, but can be written as a Prodon inequality induced by $b_W := 1$ for the sets $\{1\}, \{2\}, \{5\}, \{7\}, \{3, 5, 6\}, \{4, 6, 7\}$, and $b_W := 0$ for all other sets W in $\mathcal{F}$. So the coefficients on all depicted edges are 1, and the right-hand side is 6. Here, nodes 1 and 2 are nodes of type 2; nodes 5 and 7 are nodes of type 1; all others are of type 0. The Prodon inequality of Figure 8.1 can be proved to be facet-defining for $2\mathrm{NCON}(G; r)$, where G consists exactly of the depicted nodes and edges.

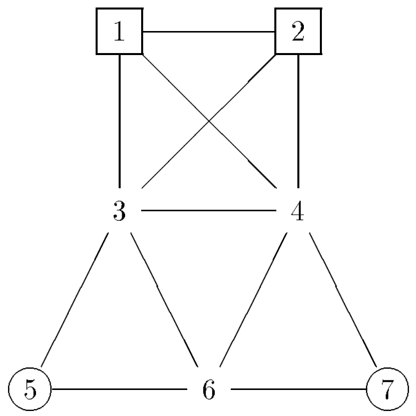


Figure 8.1: Prodon inequality

We show in the following remark that no Prodon inequality except the cut inequalities are facet-defining if there are no nodes of type 1.

(8.4) Remark. *If (G, r) is an instance of the 2ECON problem, where node types r_v only take values 0 and 2 for all $v \in V$, then no Prodon inequalities except the cut constraints define facets of 2ECON$(G; r)$.*

Proof. Let $\sum_{ij} a_{ij} x_{ij} \geq \sum_{W \in \mathcal{F}} b_W$ be a Prodon inequality. By definition,

$$a_{ij} \geq \frac{1}{2} s(\mathcal{F}; b; i; j) + \frac{1}{2} s(\mathcal{F}; b; j; i),$$

which is the same as 1/2 times the sum of all b_W over $W \in \mathcal{F}$ with $ij \in \delta(W)$. Therefore,

$$a^T x \geq \frac{1}{2} \sum_{W \in \mathcal{F}} b_W x(\delta(W)).$$

Since $x(\delta(W)) \geq \mathrm{con}(W) = 2$ for all $W \in \mathcal{F}$, this expression is at least $\sum_{W \in \mathcal{F}} b_W$ for all $x \in 2\mathrm{ECON}(G; r)$. So our Prodon inequality is implied by the sum of some cut inequalities, and must itself be a cut inequality, if it is to be facet-defining. □

Inequalities of Prodon type can similarly be defined for kECON problems, if we take as a basis the formulation of the DECON problem for node types $r_v \in \{0, 1, 2, 4, 6, \ldots\}$.

We close this section by observing that the separation problem for Prodon inequalities can be performed in polynomial time. The separation algorithm also makes use of projection and works in the same way as one iteration of Benders' decomposition method, see Benders (1962). This observation is meant to show that projection is not only of theoretical value but also of computational interest.

Suppose a point x^* with $0 \leq x^* \leq 1$ is given, for which it has to be decided, whether there is a Prodon inequality violated by this point or not. This can be decided by solving the following LP derived from (8.1):

$$\text{(8.5)} \qquad \begin{array}{lllll} & \min & z & & \\ \text{subject to} & & & & \\ \text{(i)} & y(\delta^-(W)) & +z & \geq 1 & \text{for all } W \in \mathcal{F}; \\ \text{(ii)} & y_{(i,j)} & & \geq 0 & \text{for all } (i,j) \in A; \\ \text{(iii)} & -y_{(i,j)} - y_{(j,i)} & -zx^*_{ij} & = -x^*_{ij} & \text{for all } ij \in E; \\ \text{(iv)} & & z & \geq 0 & \end{array}$$

This LP has the feasible solution $y = 0$ and $z = 1$. If its optimal value is 0, and y^* is an optimal solution, then (x^*, y^*) is feasible for the system (8.1), hence x^* satisfies all Prodon inequalities (by the projection result). If the optimal value is non-zero, then the optimal dual variables b_W for inequalities (i) and a_{ij} for equations (iii) define a Prodon inequality violated by x^*. More explicitly, the optimal dual variables b_W ($W \in \mathcal{F}$) and $a \in \mathbf{R}^E$ satisfy

$$\text{(8.6)} \qquad \begin{array}{rcll} -a_{ij} + b_W & \leq & 0 & \text{for all } ij \in E \text{ and all } W \text{ separating } i \text{ and } j \\ -a^T x^* + \sum_{w \in \mathcal{F}} b_W & > & 0. & \end{array}$$

The first inequality implies that a_{ij} is at least the maximum of $s(\mathcal{F}; b; i; j)$ and $s(\mathcal{F}; b; j; i)$ for each $ij \in E$. The optimality of a and b implies that a_{ij} is exactly the maximum of these two numbers. By multiplying a and b with some number, b can be made integral without changing (8.6). So b induces a Prodon inequality violated by x^*.

The LP (8.5) can be solved in polynomial time, since there exist polynomial separation algorithms for the directed cut inequalities (8.5)(i). Therefore, the Prodon inequalities can also be separated in polynomial time. We have, however, not made use of these inequalities yet.

REFERENCES

1. A. Agrawal, Ph. Klein and R. Ravi (1991), "When trees collide: An approximation algorithm for the generalized Steiner tree problem on networks", in the *Proceedings of the Symposium of the Theory of Computing 1991.*

2. E.Balas and W.R. Pulleyblank (1983), "The perfect matchable subgraph polytope of a bipartite graph", *Networks* **13**, 495-516.

3. M.O. Ball, W.G. Liu and W.R. Pulleyblank (1987), "Two-terminal Steiner tree polyhedra", Technical Report 87466-OR, University of Bonn.

4. Bellcore (1988), "FIBER OPTIONS: Software for designing survivable optical fiber networks", Software Package.

5. J. F. Benders (1962), "Partitioning procedures for solving mixed-variable programming problems", *Numerische Mathematik* **4**, 238-252.

6. D. Bienstock, E.F. Brickell and C.L. Monma (1990), "On the structure of minimum-weight k-connected spanning networks", *SIAM J. Disc. Math.* **3**, 320-329.

7. R. E. Bixby (1991), "Implementing the simplex method: The initial basis", Technical Report TR 90-32, Department of Mathematical Sciences, Rice University, Houston, Texas.

8. G-R. Cai and Y-G. Sun (1989), "The minimum augmentation of any graph to a k-edge connected graph," *Networks* **19**, 151-172.

9. R.H. Cardwell, C.L. Monma and T.H. Wu (1989), "Computer-aided design procedures for survivable fiber optic networks", *IEEE J. Selected Areas of Communications* **7**, 1188-1197.

10. R.H. Cardwell, T.H. Wu and W.E. Woodall (1988), "Decreasing survivable network cost using optical switches", *Proc. GLOBECOM '88.*

11. S. Chopra (1990), "Polyhedra of the equivalent subgraph problem and some edge connectivity problems", J.L. Kellogg Graduate School of Management, Northwestern University, Evanston, Illinois.

12. S. Chopra (1991), " The k-edge connected spanning subgraph polyhedron", J.L. Kellogg Graduate School of Management, Northwestern University, Evanston, Illinois.

13. W. Chou and H. Frank (1970), "Survivable communication networks and the terminal capacity matrix", *IEEE Transactions on Circuit Theory*, CT-17 (2) 192-197.

14. V. Chvátal (1973), "Edmonds polytopes and a hierarchy of combinatorial problems", *Discrete Mathematics* **4**, 305-337.

15. G. Cornuéjols, J. Fonlupt and D. Naddef (1985), "The traveling salesman problem on a graph and some related integer polyhedra", *Mathematical Programming* **33**, 1-27.

16. G. Dahl (1991), "Contributions to the design of survivable directed networks", Ph.D. Thesis, University of Oslo, Technical Report TF R 48/91, Norwegian Telecom, Research Dept., Kjeller, Norway.

17. J. Edmonds (1965), "Maximum matching and a polyhedron with 0,1-vertices, *Journal of Research of the National Bureau of Standards B* **69**, 125-130.

18. K.P. Eswaran and R.E. Tarjan (1976), "Augmentation problems", *SIAM Journal on Computing* **5**, (4) 653-665.

19. A. Frank (1990), "Augmenting graphs to meet edge-connectivity requirements", submitted to *SIAM Journal on Discrete Mathematics.*

20. A. Frank (1992), "Connectivity and network flows", Chapter 2 of the Handbook of Combinatorics, R. Graham, M. Grötschel, L. Lovász (eds.), North Holland, Amsterdam.

21. H. Frank and W. Chou (1970), " Connectivity considerations in the design of survivable networks", *IEEE Transactions on Circuit Theory*, CT-17 (4) 486-490.

22. G.N. Frederickson and J. JáJá (1982), "On the relationship between the biconnectivity augmentation and traveling salesman problem," *Theoretical Computer Science* **19**, 189-201.

23. M.R. Garey and D.S. Johnson (1979), "Computers and Intractability: A Guide to the theory of NP-completeness," Freeman, San Francisco.

24. M.X. Goemans and D.J. Bertsimas (1990), "Survivable networks, linear programming relaxations and the parsimonious property", OR Center Report OR216-90, MIT, to appear in *Mathematical Programming.*

25. M. X. Goemans (1991a), "Arborescence polytopes for series-parallel graphs", to appear in *Discrete Applied Mathematics.*

26. M. X. Goemans (1991b), "The Steiner tree polytope and related polyhedra", Department of Mathematics, MIT, Cambridge.

27. R.E. Gomory and T.C. Hu (1961), "Multi-terminal network flows", *Journal of the Society for Industrial and Applied Mathematics* **9**, 551-570.

28. M. Grötschel, L. Lovász and A. Schrijver (1988), "Geometric Algorithms and Combinatorial Optimization", Springer, Berlin.

29. M. Grötschel and C.L. Monma (1990), "Integer polyhedra associated with certain network design problems with connectivity constraints", *SIAM Journal on Discrete Mathematics* **3**, 502-523.

30. M. Grötschel, C.L. Monma and M. Stoer (1992a), "Facets for polyhedra arising in the design of communication networks with low-connectivity constraints", *SIAM Journal on Optimization* **2**, 474-504.

31. M. Grötschel, C.L. Monma and M. Stoer (1992b), "Computational results with a cutting plane algorithm for designing communication networks with low-connectivity constraints", *Operations Research* **40**, 309-330.

32. M. Grötschel, C.L. Monma and M. Stoer (1992c), "Polyhedral and computational investigations for designing communication networks with high survivability requirements", ZIB-Preprint SC 92-24, Konrad-Zuse-Zentrum für Informationstechnik Berlin, submitted to *Operations Research.*

33. M. Grötschel and M.W. Padberg (1985), "Polyhedral theory", in: E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan & D. Shmoys (eds.), *The traveling salesman problem*, Wiley, Chichester, 251-305.

34. J. Hao and J. B. Orlin (1992) "A faster algorithm for finding the minimum cut in a graph", Proceedings of the Third Annual ACM-SIAM-Symposium on Discrete Algorithms, Orlando, Florida, 165-174.

35. F. Harary (1962), "The maximum connectivity of a graph", *Proceedings of the National Academy of Sciences, USA* **48**, 1142-1146.

36. S. Khuller and U. Vishkin (1992), "Biconnectivity approximations and graph carvings", to appear in *STOC 1992*.

37. C-W. Ko and C.L. Monma (1989), "Heuristic methods for designing highly survivable communication networks", Technical report, Bellcore.

38. D.J. Kolar and T.H. Wu (1988), "A study of survivability versus cost for several fiber network architectures", *Proc. ICC '88*, 61-66.

39. E. Lawler (1976) "Combinatorial optimization: networks and matroids", Holt, Reinhart and Winston, 1976.

40. E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan & D. Shmoys (1985), *The traveling salesman problem*, Wiley, Chichester, 1985.

41. F. Ling and T. Kameda (1987), "Complexity of graph connectivity functions", Technical Report, School of Computing Science, Simon Fraser University, Burnaby, British Columbia, Canada.

42. R. Lorentzen and H. Moseby (1989), "Mathematical models and algorithms used in the subscriber network planning tool ABONETT", Norwegian Telecommunications Research Dept., TF-report 66/89.

43. L. Lovász (1976), "On some connectivity properties of Eulerian graphs", *Acta Math. Acad. Scient. Hung.* **28**, 129-138.

44. L. Lovász and M.D. Plummer (1986), "Matching Theory", *Annals of Discrete Mathematics* **29**.

45. W. Mader (1978), "A reduction method for edge-connectivity in graphs", *Ann. Discrete Math.* **3**, 145-164.

46. A.R. Mahjoub (1988), "Two edge connected spanning subgraphs and polyhedra", University of Bonn, Technical Report No. 88520-OR.

47. C.L. Monma, B.S. Munson and W.R. Pulleyblank (1990), "Minimum-weight two-connected spanning networks", *Mathematical Programming* **46**, 153-171.

48. C.L. Monma and D.F. Shallcross (1989), "Methods for designing communication networks with certain two-connected survivability constraints", *Operations Research* **37**, 531-541.

49. D. Naor, D. Gusfield and C. Martel (1990), "A fast algorithm for optimally increasing the edge-connectivity", *Proceedings of the Foundation of Computer Science '90*, 698-707.

50. C. St. J. A. Nash-Williams (1960), "On orientations, connectivity, and odd vertex pairings in finite graphs", *Canadian Journal of Mathematics* **12**, 555-567.

51. Newark Star Ledger (1987), "Damage to fiber cable hinders phone service", *Newark Star Ledger* (September 22, 1987).

52. Newark Star Ledger (1988a), "Cable snaps, snags area phone calls", *Newark Star Ledger* (February 26, 1988).

53. Newark Star Ledger (1988b), "Phone snafu isolates New Jersey; long-distance cable snaps", *Newark Star Ledger* (November 19, 1988).

54. New York Times (1988), "Phone system feared vulnerable to wider disruptions of service", *New York Times* (May 26, 1988).

55. New York Times (1989), "Experts say phone system is vulnerable to terrorists", *New York Times* (February 8, 1989).

56. M.W. Padberg and M.R. Rao (1982), "Odd minimum cut sets and b-matchings", *Mathematics of Operations Research* **7**, 67-80.

57. C.H. Papadimitriou and K. Steiglitz (1982), "Combinatorial optimization: algorithms and complexity", Prentice-Hall, Englewood Cliffs, NJ.

58. C.H. Papadimitriou, M. Yannakakis (1982), "The complexity of facets and some facets of complexity" *J. Assoc. Comp. Mach.* **29**, 285-309.

59. A. Prodon, Liebling and Gröflin (1985) "Steiner's problem on two-trees", Technical Report RO-830315, École Polytechnique Fédérale de Lausanne, Switzerland.

60. A. Prodon (1985), "A polyhedron for Steiner trees in series-parallel graphs", Technical Report, École Polytechnique Fédérale de Lausanne, Switzerland.

61. W.R. Pulleyblank (1990), "Polyhedral combinatorics", in Nemhauser, Rinnoy Kan, Todd (eds.), "Optimization" Handbook in Operations Research and Management Science, 1991.

62. A. Rosenthal and A. Goldner (1977), "Smallest augmentation to biconnect a graph", *SIAM Journal of Computing*, 6(1) 55-66.

63. A. Schrijver (1986), "Theory of linear and integer programming", Wiley, New York.

64. K. Steiglitz, P. Weiner and D.J. Kleitman (1969), "The design of minimum cost survivable networks", *IEEE Transaction on Circuit Theory* **16**, 455-460.

65. M. Stoer (1992), "Design of survivable networks", Ph.D. thesis, University of Augsburg, *Lecture Notes in Mathematics*, 1531, Springer, Heidelberg.

66. K. Takamizawa, T. Nishizeki, and N. Saito (1982), "Linear-time computability of combinatorial problems on series-parallel graphs", *Journal of the Association for Computing Machinery* **29**, (3) 623-641.

67. S. Ueno, Y. Kajitani, and H. Wada (1988), "Minimum augmentation of a tree to a k-edge-connected graph", *Networks* **18**, 19-25.

68. J.A. Wald and C.J. Colbourn (1983), "Steiner trees, partial 2-trees and minimum IFI networks", *Networks* **13**, 159-167.

69. Wall Street Journal (1988), "Fire in fiber gateway sparks flight delays, problems at brokerages", *Wall Street Journal*, (May 11, 1988).

70. T. Watanabe and A. Nakamura (1987), "Edge-connectivity augmentation problems", *Computer and System Sciences* **35**, (1) 96-144.

71. P. Winter (1985a), "Generalized Steiner tree problem in Halin networks", In *Proc. 12th Intern. Symp. on Mathematical Programming*, MIT.

72. P. Winter (1985b), "Generalized Steiner problem in outerplanar networks", *BIT* **25**, 485-496.

73. P. Winter (1986), "Generalized Steiner problem in series-parallel networks", *Journal of Algorithms* **7**, 549-566.

74. P. Winter (1987), "Steiner problem in networks: A survey", *Networks* **17**, (2) 129-167.

75. T.H. Wu and R.H. Cardwell (1988a), "Optimum routing in fiber network design: models and applications", *Proc. ICC '88.*

76. T.H. Wu, D.J. Kolar and R.H. Cardwell (1988b), "Survivable network architectures for broadband fiber optic networks: model and performance comparison", *IEEE Journal Lightwave Technology* **6**, 1698-1709.

77. G. Zorpette (1989), "Keeping the phone lines open", *IEEE Spectrum* (June 1989) 32-36.