Strengthening Integrality Gaps for Capacitated Network Design and Covering Problems

Robert D. Carr* Lisa K. Fleischer† Vitus J. Leung* Cynthia A. Phillips*

Abstract

A *capacitated covering IP* is an integer program of the form $\min\{cx|Ux \geq d, 0 \leq x \leq b, x \in Z^+\}$, where all entries of c, U, and d are nonnegative. Given such a formulation, the ratio between the optimal integer solution and the optimal solution to the linear program relaxation can be as bad as $||d||_\infty$, even when U consists of a single row. We show that by adding additional inequalities, this ratio can be improved significantly. In the general case, we show that the improved ratio is bounded by the maximum number of non-zero coefficients in a row of U, and provide a polynomial-time approximation algorithm to achieve this bound. This improves the previous best approximation algorithm which guaranteed a solution within the maximum row *sum* times optimum.

We also show that for particular instances of capacitated covering problems, including the minimum knapsack problem and the capacitated network design problem, these additional inequalities yield even stronger improvements in the IP/LP ratio. For the minimum knapsack, we show that this improved ratio is at most 2. This is the first non-trivial IP/LP ratio for this basic problem.

Capacitated network design generalizes the classical network design problem by introducing capacities on the edges, whereas previous work only considers the case when all capacities equal 1. For capacitated network design problems, we show that this improved ratio depends on a parameter of the graph, and we also provide polynomial-time approximation algorithms to match this bound. This improves on the best previous m-approximation, where m is the number of edges in the graph. We also discuss improvements for some other special capacitated covering problems, including the fixed charge network flow problem. Finally, for the capacitated network design problem, we give some stronger results and algorithms for series parallel graphs and strengthen these further for outerplanar graphs.

Most of our approximation algorithms rely on solving a single LP. When the original LP (before adding our strengthening inequalities) has a polynomial number of constraints, we describe a combinatorial FPTAS for the LP with our (exponentially-many) inequalities added. Our contribution here is to describe an appropriate separation algorithm to work in the setting of the FPTAS of Garg and Könemann. For exactly solving the LP using the ellipsoid method, we describe simpler separation routines.

1 Introduction

The US government has identified 7 areas of *critical infrastructure*, sets of services whose functioning is essential for the current operation of the country. Most of these infrastructures are physical networks such as telecommunications, water, natural gas, and transportation. The importance of network design is further evident from the wealth of literature on the subject [11]. The research in this paper is motivated by a capacitated network design problem arising in network security. However, the tools we have devised to understand this problem are applicable not only to building networks, but also to both more specific problems, such as the minimum knapsack problem, and more general problems, such as general capacitated covering problems.

The *capacitated network design problem* is defined on a multigraph $G = (V, E)$ with cost vector $c : E \to R^+$, capacity vector $u : E \to R^+$, and symmetric demand matrix $D \in \mathbf{N}^{|\mathbf{V}|^2}$, where D_{ij} is the connectivity requirement between vertices i and j. The goal is to select a minimum-cost subset of edges $F \subset E$ so that the total capacity of any cutset C of (V, F) is at least the maximum demand among all pairs of vertices that are disconnected in $(V, F \backslash C)$. In this paper, we also consider the generalization where edge e may be selected up to $b(e)$ times.

The capacitated network design problem arises as a *network reinforcement problem*. Here we are given an existing network, and for each link (edge) in the network, several reinforcement options. A *reinforcement* is a protection level of a particular strength that can be added to an edge at a certain cost. In addition, for each pair of vertices in the network, there is a specified level of protection demanded. The objective is to select a minimum-cost set of reinforcements for all the edges so that an adversary with strength less than the protection level of a particular pair of vertices cannot disconnect these vertices.

For example, suppose we have a communications network and each edge has a weight (capacity) corresponding to the cost an attacker incurs to eavesdrop on that edge. There are communications protocols (e.g. [9]) where messages are

*Sandia National Labs, Albuquerque, NM 87185. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000. Email: {bobcarr@cs.sandia.gov, vjleung@mp.sandia.gov, caphill@mp.sandia.gov}.

†Industrial Engineering and Operations Research, Columbia University, New York, NY 10027. Part of this research was done while visiting CORE, Université catholique de Louvain, Belgium. Email: lisa@ieor.columbia.edu.

broken into multiple packets and packets are sent along many different paths. In order for an eavesdropper to glean *any* information from the message, he must intercept *all* packets, and therefore must have compromised a cut between the sender and receiver. By interpreting strength and protection levels as capacities, this problem is easily reinterpreted as a capacitated network design problem.

Capacitated network design is one generalization of the minimum knapsack problem. The *minimum knapsack problem* is defined by a set of objects, each with a cost and a value, and a specified demand. The goal is to select a minimum cost set of edges with total value at least the demand. This is equivalent to capacitated network design on a graph consisting of 1 vertex pair with multiple parallel edges.

One generalization of capacitated network design is capacitated covering. A *capacitated covering IP* is an integer program (IP) of the form $\min\{cx|Ux \geq d, 0 \leq x \leq b, x \in Z^+\}$, where all entries of c, U, and d are nonnegative. To see that this is a generalization, we write the capacitated network design problem as a capacitated covering IP below. Here $x(e)$ is the number of copies of edge e we select, $\mathcal{C}$ is the set of all cutsets, and D_C is the maximum of D_{ij} over all pairs of vertices i and j disconnected by the removal of C.

$$\text{(IP1)} \qquad \begin{array}{lll} \min & \sum_{e\in E} c(e)x(e) & \\ \forall C \in \mathcal{C}: & \sum_{e\in C} u(e)x(e) & \geq \quad D_C \\ \forall e \in E: & x(e) & \in \quad \{0,1\}. \end{array}$$

Since the minimum knapsack problem is NP-hard [22], all of the abovementioned problems are NP-hard problems. In this paper, we focus on obtaining improved approximation algorithms for these problems. A *ρ-approximation algorithm* is a polynomial-time algorithm that returns a solution with cost at most ρ times the cost of the optimal solution. A *fully polynomial-time approximation scheme (FPTAS)* is an algorithm that, given $\epsilon < 1$, returns a solution of cost at most $1+\epsilon$ times the optimal solution in time polynomial in the size of the problem, and $1/\epsilon$. One special case of the capacitated network design problem is the Steiner tree problem, which is known to be MAXSNP-hard [4]. Thus we cannot hope to find an FPTAS for this problem.

Most of our approximation algorithms depend on strengthening the LP relaxation of the given IP. (The LP relaxation is the problem obtained by removing the integrality constraint on the variables.) An LP relaxation of an integer program can be *strengthened* by adding inequalities that are satisfied by all integer solutions. These inequalities are called *valid*. Given a problem instance P, we denote its optimal solution by OPT(P). For an IP, and its relaxation LP, we refer to the ratio of their optimal solutions, OPT(IP)/OPT(LP), as the IP/LP ratio.

1.1 Previous Work. The best previous approximation algorithm for the capacitated network design problem is the algorithm that greedily removes the unnecessary edges in order of decreasing cost. This finds a solution within a factor of $m := |E|$ of the optimum [12].

Many of the approximation algorithms for network design problems that achieve approximation guarantees that are better than linear consider the *uncapacitated network design problem*, where $u(e) = 1$ for all edges e, and multiple copies of each edge are allowed (although some do handle upper bounds on the number of copies of each edge). In particular, Jain [21] describes a 2-approximation for precisely this problem. Before [21], the best approximation guarantees obtained by polynomial-time algorithms for the uncapacitated network design problem were all logarithmic in $n := |V|$. For references, and a survey of related work, see for example [13].

When all edge costs are also uniform, the problem remains NP-hard, even when all demands are also uniform. The best known approximation in this case is $2 - 1/\lambda$ when connectivity requirement is λ [23]. If, in addition, the underlying graph is the complete graph and multiple edges are allowed, then the problem is solvable in polynomial time [8, 28].

Other researchers have considered approximation algorithms for the capacitated network design problem when the objective is to design a network with enough capacity to route all demands simultaneously, without any restriction on the number of copies of edges allowed [1, 6, 26, 31].

There has also been significant research on developing the techniques of integer programming and polyhedral combinatorics to attack these problems. For example, see [3, 7, 25].

The knapsack problem has been studied extensively [27], and is one of the original NP-complete problems [22]. While the knapsack problem and the minimum knapsack problems are equivalent if an exact solution is sought, they are not equivalent for approximation purposes in that a ρ-approximation algorithm for one problem does not imply the existence of a comparable guarantee for the second. The FPTAS for knapsack [24, 19] can be easily modified to work for min knapsack. However, the bound on the IP/LP ratios for the two problems is vastly different: 2 for knapsack versus D for minimum knapsack.

1.2 Our results. Almost all of the known approximation algorithms for network design problems, and many covering problems, consider a standard integer programming (IP) formulation of the problem and the optimal solution value obtained from the corresponding linear programming (LP) relaxation [13]. These algorithms then construct an integer solution and prove its approximation guarantee by comparing it with the value of the LP relaxation. One difficulty in

approximating the capacitated network design problem lies in the fact that the ratio of the optimal IP solution to the optimal LP solution can be as bad as $D_{\max} := \max_{i,j \in V^2} D_{ij}$. This holds also for the minimum knapsack problem. Thus one cannot hope to obtain improved approximation guarantees by comparing an integer solution obtained by any means to the optimal LP value.

We add a class of simple inequalities, which we call *knapsack cover (KC) inequalities*, that provably strengthen the standard linear programming relaxation. For the minimum knapsack problem, we show that the improved IP/LP ratio with these inequalities is 2. This improves on the best previous bound of $D_{\max}$. For capacitated covering problems, we add this class of inequalities for each constraint in the original IP formulation. We show that the resulting IP/LP ratio is strengthened to be bounded by the maximum number of nonzero coefficients in a row of the constraint matrix. For capacitated network design problems, we show an improved ratio that equals $\beta(G) + 1$, where $\beta(G)$, defined below, is a parameter of the graph G that is often significantly less than m, and never greater.

DEFINITION 1.1. *A* bond *is a minimal cardinality set of edges whose removal disconnects a pair of vertices with positive demand. For a multigraph G, $\beta(G)$ is the cardinality of the maximum-cardinality bond in the underlying simple graph. For example, in Figure 1, $\beta(G) = 5$.*

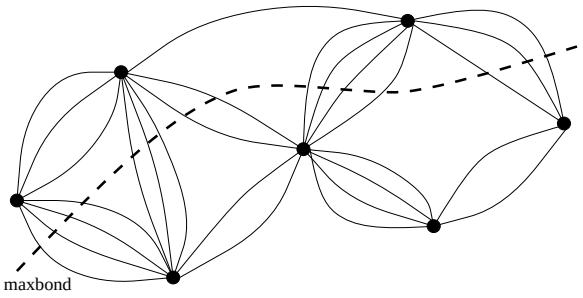

Figure 1: A bond

Series-parallel graphs are defined inductively. Each series-parallel graph has a source node s and a sink node t, called the *defining nodes*. A single edge (s, t) is the simplest series-parallel graph. New series-parallel graphs are built by composing two series-parallel graphs having sources s_1 and s_2 and sinks t_1 and t_2 respectively. In a *series* composition, the source s_2 and sink t_1 are unified. In a *parallel* composition the sources s_1 and s_2 are unified and the sinks t_1 and t_2 are unified. We can represent the series and parallel steps used to produce a graph in a *decomposition tree*. For series-parallel graphs, we show that the IP/LP ratio can be improved to $\lfloor \beta(G)/2 \rfloor + 2$, and that this bound is almost tight for the LP with KC inequalities.

We also describe polynomial-time approximation algorithms that meet these bounds. Thus, for the capacitated covering problem, our result improves the best previous approximation guarantee of the maximum row sum of A [5]. For capacitated network design, our algorithms improve the best previous approximation guarantee of m [12].

We describe some additional capacitated covering problems for which KC inequalities can be used to obtain improved approximation guarantees. These include generalized vertex cover, multicolor network design, and fixed charge network flow. The *vertex cover problem* is to select a minimum weight set of vertices so that each edge is incident to at least one selected vertex. There are several 2-approximations known for this problem. In the *multiple-choice vertex cover problem*, each vertex is actually a cluster of weighted vertices. An edge from cluster i to cluster j is *covered* if a subset of vertices from clusters i and j are selected with total weight at least the demand of the edge. The objective is again to select a minimum weight subset of vertices so that all edges are covered. Our algorithms yield a 3-approximation for this problem. If the graph is bipartite, the problem is still NP-hard, since it generalizes min knapsack. We describe a 2-approximation.

The *multi-color network design problem.* In this problem, each edge has a constant number of different types (colors) of capacities. Demand pairs come with a specified type and amount of capacity demand. The objective is to build a minimum cost network to satisfy all demands. Given p capacity types u_i, $i = 1, \ldots, p$ such that $u_i(e) > u_i(e') \Rightarrow u_j(e) \geq u_j(e')$ for all edges e and all capacity types i and j, we obtain a $\beta(G) + 1$ approximation for this problem.

The *fixed charge flow problem* has a fixed cost associated with each arc, in addition to a per-unit-flow cost. The objective is to build a network with enough capacity to route given demand between two nodes to minimize the total cost of building the network and sending the flow. For the 2-node problem, we give a 2-approximation based on introducing inequalities to tighten the LP relaxation. We also show how to model this problem as a capacitated network design problem, yielding a $\beta(G) + 1 + \epsilon$ approximation guarantee in general graphs.

Our approximation algorithms depend on solving a single LP with an exponential number of constraints. Given a polynomial-time separation oracle, this can be done in polynomial time using the ellipsoid method [15]. When the original LP (before adding our exponentially-many KC inequalities) has a polynomial number of constraints, we describe a combinatorial FPTAS for solving the strengthened LP. We do this by describing an appropriate separation algorithm required by the FPTAS for positive packing and covering described by Garg and Könemann [10]. For solving the LP using the ellipsoid method, or simplex method, we describe simpler separation routines.

When there is only one demand pair, Schwarz and Krumke [32] describe an FPTAS on series parallel graphs, if this demand pair corresponds to the defining nodes of the series parallel graph. An *outerplanar graph* is a planar graph that can be embedded so that all vertices lie on the outside face. Frequently pipeline infrastructure networks (natural gas, water) are outerplanar at the highest distribution levels. We describe an FPTAS for this problem on outerplanar graphs, without restricting the location of the demand pair.

2 Strengthening the LP

In this section, we describe inequalities to strengthen the linear programming relaxation for capacitated covering problems and show how they can be used to obtain improved approximation algorithms.

2.1 The Minimum Knapsack Problem and KC Inequalities. In this section we restrict our attention to the capacitated network design problem with demand D on two-node graphs with many parallel arcs E. This is the minimum knapsack problem. The IP/LP ratio for the minimum knapsack problem, and hence IP1, can be as large as the demand D. Consider a set with just 2 elements e_1 and e_2, and demand D. Let $u(e_1) = D-1$, $c(e_1) = 0$, $u(e_2) = D$, and $c(e_2) = 1$, where u is the vector of values of the elements. Any feasible integer solution must include element e_2 for a cost of 1, while the optimal LP solution sets $x(e_1) = 1$ and $x(e_2) = 1/D$ for a cost of $1/D$.

We introduce inequalities to strengthen the linear-programming relaxation for this problem. In general graphs, these inequalities are defined on subsets of edges corresponding to cutsets. In capacitated covering IP's, these inequalities are defined for each inequality of the IP.

Consider a set of edges A such that $u(A) < D$, and let $D(A) = D - u(A)$ be the unmet demand after selecting all edges in A. We call $D(A)$ the *residual demand*. Then any feasible solution to the minimum knapsack problem when restricted to $E \setminus A$ must also be feasible for the minimum knapsack problem on $E \setminus A$ with demand $D(A)$. As with any minimum-knapsack problem, we can assume that the capacity of each edge is no more than the demand. Let $u_A(e) := \min\{u(e), D(A)\}$. The subproblem constraint is enforced by a *knapsack cover (KC) inequality*:

$$\sum_{e \in E \setminus A} u_A(e)x(e) \geq D(A). \tag{2.1}$$

Under certain conditions, these inequalities are facet defining, see [33] for example.

Previous researchers [2, 17, 33] have considered an uncapacitated form of inequality (2.1) that forces the choice of at least one edge in E/A. We can show that if only these weaker constraints are added to IP1, the IP/LP ratio can still be as bad as $\sqrt{m}/2$. These researchers used lifting procedures to strengthen these basic inequalities, but do not specifically consider knapsack cover inequalities.

The strengthened integer program for minimum knapsack is:

$$\begin{array}{lrcl} \min & \sum_{e \in E} c(e)x(e) & & \\ \forall A \subset E \text{ with } u(A) < D: & & & \\ & \sum_{e \in E \setminus A} u_A(e)x(e) & \geq & D(A) \\ \forall e \in E: & x(e) & \in & \{0,1\}. \end{array} \tag{IP2}$$

The knapsack cover inequalities (2.1) significantly reduce the IP/LP ratio for the minimum knapsack problem.

THEOREM 2.1. *The IP/LP ratio of IP2 is at most 2.*

Proof. Let x be any feasible solution to the linear-programming relaxation of IP2. We show that $2x$ dominates a convex combination of integer knapsack covers. That is, we show there exist $0 < \alpha_1, \ldots, \alpha_r \leq 1$ with $\sum_{j=1}^{r} \alpha_j = 1$ and r feasible solutions to IP2 $x^{(1)}, x^{(2)}, \ldots, x^{(r)}$ such that $\sum_{j=1}^{r} \alpha_j x^{(j)} \leq 2x$ (componentwise). Hence, the cheapest knapsack cover in this set has cost no more than twice the cost of x.

Let $A = \{e \in E \mid x(e) \geq 1/2\}$. Selecting the edges in set A costs at most twice their contribution to the LP solution. We must now meet the residual demand $D(A) = D - u(A)$ with the remaining edges.

Let r be the least common multiple of denominators of x. We create r partial integer solutions using edges in $E \setminus A$ so that each integer solution has capacity at least $D(A)$. Each partial solution, together with A, then gives a solution for the original problem.

For each edge $e \in E \setminus A$, we create $a(e) := 2r\ x(e)$ copies of e. We partition these copies into r buckets so that each bucket has total capacity $\geq D(A)$ and no bucket contains more than one copy of each edge e. Thus each is feasible for the reduced problem. The bucketing proceeds as follows: Reindex the edges in $E \setminus A$ by decreasing original capacities, so that $u(e_i) \geq u(e_j)$ for all $i < j$. Put the $a(e_1)$ copies of the first edge e_1 into the first $a(e_1)$ buckets, the $a(e_2)$ copies of edge e_2 into the next $a(e_2)$ buckets modulo r and so on.

Since we consider only edges in the set $E \setminus A = \{e \in E \mid x(e) < 1/2\}$, there are less than r copies of any one edge, and hence no edge appears more than once in any bucket.

The difference in u_A capacity between the bucket with the most capacity (the first bucket) and the least capacity (the last bucket) is at most $D(A)$: In the worst case, the first bucket b_1 contains one more edge than the last bucket b_r. Pair the i^{th} edge added to b_r with the $i+1^{st}$ edge added to b_1. By construction, in each pair the edge in b_r has at least as much capacity as its mate in b_1. Since if $u(e_1) \geq u(e_2)$ then $u_A(e_1) \geq u_A(e_2)$ for any choice of A, this also holds when comparing u_A capacities. Hence the difference in u_A

capacities of the two buckets is at most the u_A-capacity of the highest capacity edge, which is at most $D(A)$.

If one of the buckets corresponds to an infeasible solution, it has capacity $< D(A)$. Assume the last (lowest-capacity) bucket has capacity less than $D(A)$. This implies that the first bucket has u_A-capacity less than $2D(A)$, and the total u_A-capacity in all buckets is less than $2rD(A)$. This then implies that $2r \sum_{e \in E \setminus A} u_A(e)x(e) < 2rD(A)$, which contradicts the fact that x satisfies the knapsack cover inequality for set A. Hence all buckets have capacity at least $D(A)$. ■

This proof actually shows we do not need to satisfy all knapsack cover inequalities for the theorem to hold: as long as a fractional solution satisfies the demand, and the knapsack cover inequality holds for $A = \{e \in E | x(e) \geq 1/2\}$, then the value of this solution is within 1/2 of the integer optimal. We call a fractional solution x that satisfies the demand and the knapsack cover inequality for $A_\gamma = \{e \in E | x(e) \geq \gamma\}$ a *γ-integer-valued solution*. Using the ellipsoid method in the framework of [15], and the separation algorithm that checks the KC inequality for this single set A_γ, we can find a γ-integer-valued solution in polynomial time. (Note that the problem of finding a γ-integer-valued solution is not a linear program, nor even necessarily a convex program, and yet the results in [15] imply that we can still find such a solution in polynomial time.) In Section 3, we describe a combinatorial FPTAS for the LP, thus avoiding use of the ellipsoid method.

Given a fractional solution that meets the demand and satisfies the KC inequality for A, we can recover a feasible integer solution within twice its value in polynomial time. The key observation is that although the bucketing algorithm conceptually uses r buckets, the actual number of different integer solutions is at most $m+1$, where m is the number of parallel edges.

THEOREM 2.2. *There is a polynomial-time 2-approximation algorithm for the minimum knapsack problem based on rounding the linear program relaxation of IP2.*

Proof. We must identify the bucket with the cheapest feasible solution. We prove by induction that after adding copies of k edges to the buckets, we have created at most $k+1$ different integer solutions, and buckets containing the same integer solution are consecutive. Adding copies of the first edge e_1 to the first $a(e_1)$ buckets creates at most 2 different partial integer solutions, and identical integer solutions are consecutive. After adding the copies of the i^{th} edge, we have by induction $i+1$ different partial solutions, and identical integer solutions are consecutive. We add a copy of the $i+1^{st}$ edge to the bucket immediately after the bucket containing the last copy of the i^{th} edge, and continue with the remaining copies. We need only check solutions that were the same before adding edge $i+1$ but are different afterward. The first bucket to which we add edge $i+1$ already differs from its earlier neighbor in that it does not contain edge i, so that we do not create a new distinction here. However, the last bucket to which we add edge $i+1$ may have been identical to its successor. Thus we create at most one new distinct integer solution.

We keep track of distinct integer solutions by noting the $\leq m+1$ different end points of intervals of length $a(e_i)$ on a modular clock of size r. This takes linear time. A simple m^2 algorithm checks the cost of each solution and picks the cheapest. ■

The minimum knapsack problem can be solved in pseudopolynomial time using dynamic programming which readily suggests an FPTAS [24, 19]. However, unlike the FPTAS, our results are readily applicable to more general problems for which no strong approximation results exist.

The following example shows that Theorem 2.2 is almost tight. Consider the problem with m edges, each of capacity $m-1$ and cost 1, and demand m. The optimal IP solution picks any two edges for a cost of 2. The optimal LP solution assigns a value of $1/(m-1)$ to each edge, for a total cost of $m/(m-1)$. Thus the ratio of IP to LP solutions is $2 - \frac{2}{m}$.

2.2 General upper bounds on variables. All of the results in this paper extend to the setting where edge e may be selected up to $b(e)$ times. We can assume $b(e)$ is bounded by $\lceil D_{\max}/u(e) \rceil$. The knapsack cover inequalities (2.1) remain valid with a modified definition of edge-set capacities: $u(A) := \sum_{e \in A} u(e)b(e)$. For the minimum knapsack problem, it is now sufficient to satisfy these modified KC inequalities for $A := \{e \in E | x(e) \geq b(e)/2\}$. When bucketing, we create $a(e) := 2rx(e)$ copies of e. Since $2x(e) < b(e)$, no more than $b(e)$ copies of edge e occur in any bucket. The earlier arguments show that each bucket is feasible. This implies Corollary 2.3.

COROLLARY 2.3. *There is a polynomial-time 2-approximation algorithm for the minimum knapsack problem with general upper bounds using linear-programming techniques.*

The same technique works for all problems discussed in this paper, since all our approximation guarantees are insensitive to multiple edges. Hence while our proofs discuss the 0/1 problem, the theorems hold for general upper bounds.

2.3 Cyclic multi-graphs and multicommodity demands. In this section we show how knapsack cover inequalities strengthen the LP relaxation of more general instances of capacitated network design. In general graphs, knapsack

cover inequalities hold for any cut that separates demand pairs. Such a cut can be considered in isolation as a minimum knapsack problem with the cutset as the element set and the demand equal to the maximum demand separated by the cut.

A *cyclic multigraph* is a multigraph for which the underlying simple graph is a cycle.

THEOREM 2.4. *The IP/LP ratio of IP1 with knapsack cover inequalities on a cyclic multigraph is* ≤ 3.

Proof. Let x be a feasible LP solution. We show that $3x$ dominates a convex combination of feasible integer solutions. Given x, we run the bucketing algorithm on each multiedge separately, for the set of edges $\{e \in E \mid x(e) < 1/3\}$, and then take the n sets of r buckets (where n, the number of nodes in the graph, is also the number of edges in the underlying simple cycle), and merge these n sets in any order to obtain r solutions. Each integer solution contains exactly one bucket from each of the n sets. Consider a particular cutset C defined by two multiedges, with maximum demand D separated by the implied cut. Let $A = \{e \in E \mid x(e) \geq 1/3\}$, $A_C = A \cap C$ and consider the buckets for either one of the multiedges before the merge. As shown in the proof of Theorem 2.1, the difference in u_A-capacity between the highest-capacity bucket and the lowest is at most $D(A_C)$. Hence, once the buckets are merged, the difference in the total capacity across this cut between the highest capacity solution and the lowest is at most $2D(A_C)$. Thus, among all integer solutions, if the one with the lowest capacity across this cut has capacity less than $D(A_C)$, then the highest has capacity less than $3D(A_C)$, and this contradicts the fact that the knapsack cover inequality for A across the cut C is satisfied. ■

The best previous approximation bound for this problem was m [12]. Using Theorem 2.4, we can significantly improve this bound:

COROLLARY 2.5. *There exists a polynomial-time 3-approximation algorithm for the capacitated network design problem on cyclic multigraphs.*

Proof. Cyclic multigraphs have only $\binom{n}{2}$ minimal cutsets in the underlying simple graph. We can thus find a $1/3$-integer-valued solution in polynomial time using the ellipsoid method and checking the knapsack cover inequality for the set $\{e \in E \cap C \mid x(e) \geq 1/3\}$ for each cutset C. In Section 3, we show how to obtain an ϵ-approximate solution to our LP using combinatorial methods in polynomial time. This gives us a fully combinatorial polynomial-time approximation algorithm with comparable guarantee.

We note that the bucketing algorithm again produces at most $m + 1$ integer solutions. Merging two sets of $m + 1$ buckets pairwise does not create any new buckets (solutions). ■

The IP/LP ratio for the cyclic multigraph improves when there is only one pair of nodes with demand, though algorithmically we would prefer the FPTAS of [32] for series-parallel graphs.

COROLLARY 2.6. *The IP/LP ratio for the* (s, t) *capacitated network design problem on cyclic multigraphs is at most* 2.

Proof. Perform the bucketing separately for each multiedge as described in Theorem 2.4 except include all edges with $x(e) < 1/2$. The underlying cycle has exactly two disjoint paths between s and t. Let R be the set of multiedges corresponding to one such path and let $L = E - R$. All multiedges in R merge their i^{th} bucket into the i^{th} integral solution. All multiedges in L merge their i^{th} bucket into the $r - i^{th}$ integral solution. Consider a cutset C implied by a simple cut separating s and t so C contains exactly one member of R and one member of L. Let $A = \{e \in C \mid x(e) \geq 1/2\}$. The largest difference in u_A-capacities among the buckets for either multiedge is $D(A)$. When we merge the buckets for these two multiedges as described above, we create a new set of r buckets such that the difference in capacity between any 2 buckets is still at most $D(A)$. Suppose that before merging bucket i for the multiedge in R has greater capacity than bucket j. Then, for the multiedge in L, bucket i will have capacity no larger than bucket j. Thus merging the two sets of buckets cannot increase the maximum difference. The previous arguments yield the bound of 2. ■

2.4 General graphs. In this section, we extend the ideas in the proofs of Theorems 2.1 and 2.4 to obtain new bounds for general graphs. We also describe how to separate the knapsack cover inequalities for general graphs, since in general there will be exponentially-many interesting cuts. Recall the max bond $\beta(G)$ of Definition 1.1.

THEOREM 2.7. *The IP/LP ratio of IP1 with knapsack cover inequalities for a general graph* G *is at most* $\beta(G) + 1$.

Proof. Can be obtained from the proof of Theorem 2.4, by replacing 2 with $\beta(G)$ and 3 with $\beta(G) + 1$. For a cut with $\beta(G)$ multiedges, the maximum difference in capacity between two buckets (integer solutions) is $\beta(G)$ times the residual demand. ■

As before, it is sufficient to check that the knapsack inequality for $A := \{e \in C | x(e) \geq \frac{1}{\beta(G)+1}\}$ is satisfied for all cutsets C separating demand pairs. However, now there may be an exponential number of such cutsets, so this does not lead to a polynomial-time separation algorithm. Instead, we use the bucketing procedure to identify a violated knapsack inequality. Given a fractional solution x, we build the set of candidate integer solutions. We need only

check the lowest-capacity integer solution (last bucket) for feasibility. This can be done by using the polynomial-time Gomory-Hu algorithm [14] to determine the value of the minimum cut separating each pair of vertices, and comparing these values with the demand values. If some cut C has insufficient capacity, the set $\{e \in C | x(e) \geq \frac{1}{\beta(G)+1}\}$ yields a violated knapsack cover inequality.

Our algorithm as stated uses $\beta(G)$ explicitly; but computing $\beta(G)$ is NP-hard. We don't actually need to know the exact value of $\beta(G)$. Instead, we can easily determine a lower bound on β, and run the algorithm with this value. If the algorithm creates an infeasible integer solution, we check the associated knapsack cover inequality. If it is not violated, then β is too small. We can use binary search to find the smallest β which yields feasible integer solutions. This gives a $\beta + 1$-approximate solution for some $\beta \leq \beta(G)$.

2.5 Series-Parallel graphs. The special structure of series-parallel graphs reduces the IP/LP ratio. The FPTAS of [32] does not apply because we have multicommodity demands.

THEOREM 2.8. *The IP/LP ratio of IP1 with KC inequalities for a series-parallel graph G is at most $\lfloor \frac{\beta(G)}{2} \rfloor + 2$.*

Proof Hint: As in the proof of Corollary 2.6, order the buckets of multiedges in either increasing or decreasing order. By using the series-parallel decomposition tree, we can choose an orientation for each multiedge such that each bond has half its multiedges oriented in each direction. ■

This bound for the IP/LP ratio is nearly tight, since there is a series-parallel example with ratio $\lfloor \beta(G)/2 \rfloor + 1$.

2.6 The Capacitated Covering Problem. Recall, a *capacitated covering IP* is an integer program of the form $\min\{cx | Ux \geq d, 0 \leq x \leq b, x \in Z^+\}$, where all entries of c, U, and d are nonnegative. Let p be the maximum number of nonzero entries in a row of U and let q be the maximum row sum of U. If U is a $0-1$ matrix, $p = q$, but in general $p \leq q$. Hall and Hochbaum [16] give a p-approximation for U a 0-1 matrix. Bertsimas and Vohra [5] extend this to give a q-approximation for the general problem. In this section, we describe a p-approximation for general, nonnegative matrices U, improving the best previous bound. We obtain our guarantee by strengthening the LP relaxation using knapsack cover inequalities.

By the discussion in Section 2.2, we can restrict our discussion to the case when $b \equiv 1$. We treat each constraint as a minimum knapsack problem and introduce knapsack cover inequalities of the following form. Let E denote the set of all variables. Define $D_i(A) := \max\{0, d_i - \sum_{j \in A} U_{i,j}\}$ and $u_A(i,j) := \min\{U_{i,j}, D_i(A)\}$. The KC inequalities for the i^{th} constraint are $\sum_{j \in E \setminus A} u_A(i,j) x_j \geq D_i(A)$.

THEOREM 2.9. *The IP/LP ratio for the capacitated covering problem with KC inequalities is $\leq p$.*

Proof. Let x^* be a solution to the LP with KC inequalities, and let $A = \{j | x^*_j \geq 1/p\}$. Define the integer solution y by $y_j = 1$ if $j \in A$, else $y_j = 0$. Since $y \leq px^*$, $c \cdot y \leq pc \cdot x^*$. It remains to show y is a feasible solution. If y is not feasible, it violates some constraint: $U_i \cdot x \geq d_i$; that is, $D_i(A) > 0$.

Define $B := \{j \in E \setminus A \mid u_A(i,j) > 0\}$. Since $u_A(i,j) \leq D(A)$, for all $j \in B$, $u_A(i,j) \cdot x^*(j) < D_i(A)/p$. But there are at most p elements in B, implying

$$\sum_{j \in E \setminus A} u_A(i,j) x^*(j) < D_i(A)$$

which contradicts x^* satisfying all KC inequalities. ■

As with previous problems, it suffices here to find a $1/p$-integer-valued solution (a solution satisfying the KC inequalities for the set $A = \{j | x^*_j \geq 1/p\}$), which can be done in polynomial time. We can also obtain an ϵ-approximate solution to the LP via a combinatorial FPTAS as described in Section 3.

2.7 Generalized Vertex Cover. Using knapsack cover inequalities for each constraint corresponding to an edge in the generalized vertex cover problem, we show that the IP/LP ratio for this formulation is at most 3, and if the graph is bipartite, this ratio is at most 2. Polynomial-time approximation algorithms follow using previous arguments.

The 3-approximation is apparent from the fact that every constraint in the integer programming formulation of this problem involves node variables from only 2 groups of nodes. A group of nodes for this IP is analogous to a group of parallel edges for the cyclic multigraph problem. Thus, Theorem 2.4 implies a 3-approximation for this problem. If G is bipartite, we obtain a 2-approximation by using arguments similar to those in the proof of Corollary 2.6

2.8 Fixed Charge Network Flow.

We start by considering the two node graph with multiple arcs between the nodes. Each arc has a fixed cost $c(e)$ and a per unit flow cost $f(e)$. We wish to select a set of arcs with sufficient capacity to route demand D to minimize the fixed cost of the arcs, plus the cost of routing D units of flow on these arcs.

This 2-node fixed charge network flow problem can be modelled as an MIP, given below. As with the min knapsack problem, the integrality gap for (MIP1) can be quite large. We introduce a modification of the knapsack cover inequalities for this problem and show that adding these to (MIP1) reduces the integrality gap to 2. We then show how to obtain a 2-approximation algorithm using the ideas in this proof. The 2-node MIP can be generalized to arbitrary graphs and pairwise demands as in IP1.

$$\min \quad \sum_{e\in E}[c(e)x(e)+f(e)y(e)]$$

$$\text{(MIP1)} \qquad \sum_{e\in E} f(e)y(e) \;\geq\; D$$
$$\forall e\in E: \quad 0 \;\leq\; y(e) \;\leq\; u(e)x(e)$$
$$\forall e\in E: \quad x(e) \;\in\; \{0,1\}.$$

Let A be an edge set such that $D-u(A)>0$. Then for every partition $\{E_1,E_2\}$ of the edge set, the following inequalities, which we call *flow cover inequalities* are valid.

$$\text{(2.2)} \sum_{e\in E_1\setminus A} y(e) + \sum_{e\in E_2\setminus A} u_A(e)x(e) \geq D-u(A),$$
$$\forall\ D-u(A)>0,\ E_1\cup E_2=V.$$

A different form of flow cover inequalities were introduced in a polyhedral study of fixed charge problems [29]. They presented them as packing, not covering, inequalities; and they did not consider their effect at tightening the IP/LP ratio.

THEOREM 2.10. *The integrality gap for 2-node fixed-charge network flow with flow cover inequalities is 2.*

Proof. Let x,y be the optimal solution to the LP with flow cover inequalities. We will show that $2x, 2y$ dominates a convex combination of feasible solutions. Thus the minimum cost solution in this combination has cost at most twice the LP value. Let $A:=\{e|y(e)\geq u(e)/2\}$. We include all $e\in A$ in all integer solutions, and set the flow value on such e equal to $u(e)$. This at most doubles the contribution of A to the objective function.

Suppose there is an edge $e\in E\setminus A$ with $x(e)\geq 1/2$ and $y(e)\geq u_A(e)/2 = D(A)/2$ (since $y(e)\geq u(e)/2$ would imply $e\in A$). Then setting $x(e)=1$ and $y(e)=u_A(e)$ together with the set A yields a feasible integer solution of cost at most twice the LP value, and we are done. In the remaining proof, we assume this is not the case.

To distribute the remaining edges, we partition E into two sets. Let $E_1:=\{e|y(e)<u_A(e)x(e)\}$, $E_2:=E\setminus E_1$. Let r be the least common multiple of denominators of x and y/u_A. We create r feasible solutions. For edges $e\in E_1\setminus A$ we create $2ry(e)/u_A(e)$ copies. For edges $e\in E_2\setminus A$ we create $2rx(e)$ copies. The total number of copies of any edge does not exceed r. For this to hold, edges in $E_1\setminus A$ must have $y(e)/u_A(e)\leq 1/2$. Suppose this is not the case. Then by the assumption above, $x(e)<1/2$ and we have $y(e)/u_A(e)\leq x(e)$ by the definition of edge set $E_1\setminus A$. Using a similar argument, we have $x(e)\leq 1/2$ for all edges in $E_2\setminus A$. Order the edges by nonincreasing values of $u(e)$ and perform the bucketing algorithm. If edge e is placed in a bucket, we associate with this edge the corresponding values $x'(e)=1$, $y'(e)=u_A(e)$. The total cost of all r solutions obtained in this manner is

$$\begin{aligned}
&\textstyle\sum_{e\in E_1\setminus A}[2ry(e)/u_A(e)](f(e)y'(e)+c(e)) \\
&\quad + \textstyle\sum_{e\in E_2\setminus A}[2rx(e)](f(e)y'(e)+c(e)) \\
&\leq\ 2r[\textstyle\sum_{e\in E_1\setminus A}(f(e)y(e)+c(e)x(e)) \\
&\quad + \textstyle\sum_{e\in E_2\setminus A}(f(e)y(e)+c(e)x(e))] \\
&\leq\ 2r\textstyle\sum_{e\in E\setminus A}(f(e)y(e)+c(e)x(e))
\end{aligned}$$

which is at most $2r$ times the contribution of edges in $E\setminus A$ to the LP solution value. Hence, at least one of the r solutions has cost at most twice the LP value.

We must now show all solutions are feasible. Let $\gamma(e):=\min\{y'(e),u_A(e)x'(e)\}$. In words, $\gamma(e)$ is the contribution of edge e to the flow cover inequality corresponding to E_1, E_2, and A. Note that $\gamma(e)\equiv u_A(e)$. Our first observation is that the difference in $\gamma(e)$-capacity of any two buckets is at most $D-u(A)$. This holds, as before, by the ordering of the edges in the bucketing algorithm, and since $u(e_1)\geq u(e_2)$ implies $u_A(e_1)\geq u_A(e_2)$. If some bucket has $\gamma(e)$ capacity less than $D-u(A)$, then the last bucket does, since this has the least $\gamma(e)$-capacity. If the $\gamma(e)$-capacity of the last bucket is less than $D-u(A)$, then all buckets have $\gamma(e)$-capacity less than $2(D-u(A))$. Summing over the $\gamma(e)$-capacity of all copies of edges put in buckets, this then implies the following inequality:

$$\sum_{e\in E_1\setminus A} 2ry(e) + \sum_{e\in E_2\setminus A} 2ru_A(e)x(e) < 2r(D-u(A))$$

This contradicts (x,y) a solution to the LP with flow cover inequalities. Hence all buckets contain feasible solutions. ■

To obtain a 2-approximation algorithm, we need to be able to separate these inequalities. Notice that the proof of Theorem 2.10 relies only on the fact that the flow cover inequality is satisfied for the set $A=\{e|y(e)\geq u(e)/2\}$. Thus we can perform separation in the same manner as with the knapsack cover inequalities. Similarly, we can also easily obtain a approximate solution in polynomial time.

For general graphs with one demand pair, we can model the fixed charge network flow problem as a capacitated network design problem. We introduce multiple copies of edges, one for each flow amount. This greatly increases the size of the graph. Since we can only hope for an approximate solution anyway, we can control the size of this increase by adding edges for flow quantities that are a power of $(1+\epsilon)$ for $\epsilon>0$. Then our approximation algorithms for capacitated network design applied to this problem yield approximation guarantees that increase by a factor of ϵ.

3 Solving the LP Efficiently

In this section, we show how to find good solutions to capacitated covering LP's in polynomial time without using the ellipsoid method, provided the initial formulation (without KC

inequalities) contains a polynomial number of constraints. In the capacitated network design setting, this corresponds to graphs with a polynomial number of interesting cuts.

To solve LPs with many constraints, as is the case here, it may be more efficient to use a fast approximation algorithm instead of the simplex method, ellipsoid method, or other separation-based, exact solution procedure. There exist polynomial-time, combinatorial methods for approximately solving LPs with special structure using separation oracles. For example, Plotkin, Shmoys, and Tardos [30] describe such a method for linear programs with all coefficients non-negative, and all inequalities $\leq$ (a packing LP), or all inequalities $\geq$ (a covering LP). Recently, Garg and Könemann [10] describe a similar, but simpler procedure. This finds ϵ-approximate solutions to both the primal and dual problems, and relies on an oracle to find a most violated inequality, or an ϵ-approximate most violated inequality of the covering problem. For the system

$$\min\{cx|Ax \geq d, x \geq 0\}$$

and a vector $\hat{x}$, a *most violated inequality* is the row ℓ minimizing $A_\ell \hat{x}/d_\ell$.

We first consider the minimum knapsack problem. The arguments extend to more general capacitated covering problems by examining the set of KC inequalities for each original covering constraint.

For LP2, the LP relaxation of IP2, we must find an inequality minimizing $\sum_{e \in E \setminus A} \frac{u_A(e)}{D(A)} x(e)$ over all choices of A with $D(A) > 0$. The KC inequalities corresponding to $A := \{e \in E | x(e) \geq 1/2\}$, which we used previously, are not in general the most violated KC inequalities. Indeed, it is not hard to construct examples for which using them in the Garg-Könemann algorithm instead of most violated inequalities will not lead to ϵ-approximate LP solutions.

To find a most-violated inequality for the minimum knapsack problem, we fix d to a value in $[1, D]$, and search for a set of edges A that satisfies $D(A) = d$ and that minimizes $\sum_{e \in E \setminus A} \frac{u^d(e)}{d} x(e)$, where $u^d(e) := \min\{u(e), d\}$. For fixed d, each edge has a weight $w(e) := \frac{u^d(e)}{d} x(e)$ and a capacity $u(e)$. We seek a minimum-weight subset of edges of total capacity at least $T - d$, where $T := \sum_{e \in E} u(e)$. This is exactly a minimum knapsack problem, for which there exists an FPTAS [19].

We now reduce the number of calls to the knapsack cover FPTAS from D to something polynomial in the input size. Let A^* denote the set of edges that yields the most violated inequality. Let $d^* = D(A^*)$. Suppose now that we don't look for violated inequalities for all values of $d = D(A)$ in the set $[1, D]$, but only for a selected subset. If there is a $d \leq d^*$ in our set such that $\sum_{e \in E \setminus A^*} \frac{u^d(e)}{d} x(e) \leq (1 + \epsilon) \sum_{e \in E \setminus A^*} \frac{u^{d^*}(e)}{d^*} x(e)$, then the most violated inequality we find when we fix d will be at least this violated, and hence an ϵ-approximate most violated inequality. Since for $d \leq d^*$, $\frac{u^d(e)}{d} \geq \frac{u^{d^*}(e)}{d^*}$ and this inequality is least tight when $u^d(e) = u^{d^*}(e)$, we have that $\frac{u_d(e)}{d} \leq (1 + \epsilon) \frac{u^{d^*}(e)}{d^*}$ whenever $d \geq d^*/(1 + \epsilon)$. Thus, to find an ϵ-approximate most violated inequality, it suffices to restrict our search to values of d in $\{\lceil (1 + \epsilon)^i \rceil \mid i = 0, 1, \ldots, \lceil \log_{1+\epsilon} D \rceil\}$. This reduces the number of iterations of the dynamic program from D to $\frac{1}{\epsilon} \log D$.

The Garg and Könemann's FPTAS for solving positive packing LPs requires $O(\epsilon^{-2} m)$ most-violated-inequality-subroutine calls. Thus the corresponding FPTAS for solving our LP requires $O(\epsilon^{-3} m (\log D) K(m, \epsilon))$ time, where $K(m, \epsilon)$ is the time required to obtain an ϵ-approximate solution to a minimum knapsack problem on m items. For capacitated covering problems, this runtime is multiplied by the number of original covering constraints.

4 Outerplanar Graphs

In this section, we give a pseudopolynomial-time dynamic program and corresponding FPTAS for the (s, t)-capacitated network design problem on outerplanar multigraphs. This problem is NP-complete since it generalizes the minimum knapsack problem. Schwarz and Krumke [32] give a FPTAS for the (s, t) capacitated network design problem on series parallel graphs when s and t are the two endpoints. Their algorithm uses dynamic programming, moving up the series-parallel decomposition tree. Our result is not subsumed by this algorithm since we allow s and t to be any two nodes on an outerplanar graph (where such decomposition trees do not exist).

The dynamic program for outerplanar graphs proceeds as follows. Consider an outerplanar embedding of the graph G from which the multigraph is derived. If s and t are not biconnected, the problem easily partitions into two or more smaller parts whose solutions are easily combined into a solution for the whole. Therefore, we can assume without loss of generality that s and t are biconnected. Furthermore, the biconnected components not containing both s and t can be ignored.

Consider the biconnected component B containing both s and t. Since G is outerplanar, B must be a cycle C with chords. If the edge (s, t) exists, B is series-parallel, and we can use the dynamic program and FPTAS of Schwarz and Krumke. Otherwise, the vertices s and t partition $C - \{s, t\}$ into an upper path P and a lower path Q.

In general it is challenging to find an order in which to process the vertices for standard dynamic programming. We transform the graph to an equivalent form where dynamic programming is easy. Replace each vertex v_i (w_j) in P (Q) adjacent to vertices in both P and Q by a path P_i (Q_j) of length equal to the number of vertices in Q (P) to which v_i (w_j) is adjacent. Give the edges in P_i (Q_j) capacity D and

cost zero. Each vertex in $P_i(Q_j)$ becomes an endpoint of an edge originally adjacent to $v_i(w_j)$. There is only one way to assign edges to vertices while preserving outerplanarity.

The dynamic program processes each edge (v_i, w_j) (where $v_i \in P$ and $w_j \in Q$) in order from s to t. For each such edge, we compute a minimum-cost solution that routes d_i demand to v_i and $D - d_i$ demand to w_j for $d_i = 0, \ldots, D$.

References

[1] Baruch Awerbuch and Yossi Azar. Buy-at-bulk network design. In *38th Annual IEEE Symposium on Foundations of Computer Science*, pages 542–547, 1997.

[2] Egon Balas. Facets of the knapsack polytope. *MP*, 8:146–164, 1975.

[3] F. Barahona. Network design using cut inequalities. *SJO*, 6:823–837, 1996.

[4] M. Bern and P. Plassmann. The Steiner problem with edge lengths 1 and 2. *IPL*, 32:171–176, 1989.

[5] Dimitris Bertsimas and Rakesh Vohra. Rounding algorithms for covering problems. *Math. Programming*, 80:63–89, 1998.

[6] Moses Charikar, Chandra Chekuri, Ashish Goel, Sudipto Guha, and Serge Plotkin. Approximating a finite metric by a small number of tree metrics. In IEEE [20].

[7] Geir Dahl and Mechthild Stoer. A cutting plane algorithm for multicommodity survivable network design problems. *INFORMS Journal on Computing*, 10(1):1–11, 1998.

[8] A. Frank. Augmenting graphs to meet edge connectivity requirements. *SIAM J. Discr. Math.*, 5(1):25–53, 1992.

[9] Matthew Franklin. *Complexity and Security of Distributed Protocols*. PhD thesis, Dept. of Computer Science, Columbia University, February 1994.

[10] N. Garg and J. Könemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. In IEEE [20], pages 300–309.

[11] F. Glover, D. Klingman, and N. V. Phillips. *Network Models in Optimization and Their Applications in Practice*. John Wiley & Sons, Inc., New York, 1992.

[12] M. X. Goemans, A. V. Goldberg, S. Plotkin, D. Shmoys, É. Tardos, and D. P. Williamson. Improved approximation algorithms for network design problems. In *Proc. 5th Annual ACM-SIAM Symp. on Discrete Algorithms*, pages 223–232, 1994.

[13] Michel X. Goemans and David P. Williamson. The primal-dual method for approximation algorithms and its application to network design problems. In Hochbaum [18], pages 144–191.

[14] R. E. Gomory and T. C. Hu. Multi-terminal network flows. *J. Soc. Indust. Appl. Math*, 9(4):551–570, 1991.

[15] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, 1988.

[16] Nicholas G. Hall and Dorit S. Hochbaum. A fast approximation algorithm for the multicovering problem. *Discrete Applied Mathematics*, 15:35–40, 1986.

[17] Peter L. Hammer, Ellis L. Johnson, and Uri N. Peled. Facets of regular 0-1 polytopes. *MP*, 8:179–206, 1975.

[18] D. S. Hochbaum, editor. *Approximation Algorithms for NP-Hard Problems*. PWS Publishing Company, Boston, 1997.

[19] O. H. Ibarra and C. E. Kim. Fast approximation algorithms for the knapsack and sum of subset problems. *JACM*, 22:463–468, 1975.

[20] *39th Annual Symposium on Foundations of Computer Science*, 1998.

[21] Kamal Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. In IEEE [20].

[22] Richard M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[23] Samir Khuller. Approximation algorithms for finding highly connected subgraphs. In Hochbaum [18], pages 236–265.

[24] Eugene L. Lawler. Fast approximation algorithms for knapsack problems. *Mathematics of Operations Research*, 4:339–356, 1979.

[25] Thomas L. Magnanti, P. Mirchandani, and R. Vachani. Modeling and solving the two-facility capacitated network loading problem. *Operations Research*, 43(1):142–157, 1995.

[26] Y. Mansour and D. Peleg. An approximation algorithm for minimum-cost network design. Technical Report CS94-22, The Weizman Institute of Science, Rehovot, Isreal, 1994.

[27] Silvano Martello and Paolo Toth. *Knapsack Problems: Algorithms and computer implementations*. Series in Discrete Mathematics and Optimization. Wiley-Interscience, 1990.

[28] D. Naor, D. Gusfield, and C. Martel. A fast algorithm for optimally increasing the edge connectivity. In *Proc. 31st Annual IEEE Symposium on Foundations of Computer Science*, pages 698–707, 1990.

[29] M. W. Padberg, T. J. Van Roy, and L. A. Wolsey. Valid inequalities for fixed charge problems. *Oper. Res.*, 33:842–861, 1985.

[30] S. A. Plotkin, D. Shmoys, and É. Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research*, 20:257–301, 1995.

[31] F. S. Salman, J. Cheriyan, R. Ravi, and S. Subramanian. Buy-at-bulk network design: approximating the single-sink edge installation problem. In *Proc. 8th ACM-SIAM Symp. on Discrete Algorithms*, 1997.

[32] S. Schwarz and S. O. Krumke. On budget constrained flow improvement. *Information Processing Letters*, 66:291–297, 1998.

[33] Laurence A. Wolsey. Facets for a linear inequality in 0-1 variables. *Mathematical Programming*, 8:168–175, 1975.