An Iterative Rounding 2-Approximation Algorithm for the Element Connectivity Problem

Lisa Fleischer
GSIA
Carnegie Mellon University
5000 Forbes Ave.,
Pittsburgh, PA 15217
lkf@andrew.cmu.edu

Kamal Jain
Microsoft Research
One Microsoft Way
Redmond, WA 98052
kamalj@microsoft.com

David P. Williamson
IBM Almaden Research Center
650 Harry Rd.
San Jose, CA 95120
dpw@almaden.ibm.com

Abstract

In the survivable network design problem (SNDP), given an undirected graph and values r_{ij} for each pair of vertices i and j, we attempt to find a minimum-cost subgraph such that there are r_{ij} disjoint paths between vertices i and j. In the edge connected version of this problem (EC-SNDP), these paths must be edge-disjoint. In the vertex connected version of the problem (VC-SNDP), the paths must be vertex disjoint. Jain et al. [12] propose a version of the problem intermediate in difficulty to these two, called the element connectivity problem (ELC-SNDP, or ELC). In this problem, the set of vertices is partitioned into terminals and nonterminals. The edges and nonterminals of the graph are called elements. The values r_{ij} are only specified for pairs of terminals i, j, and the paths from i to j must be element disjoint. Thus if $r_{ij} - 1$ elements fail, terminals i and j are still connected by a path in the network.

These variants of SNDP are all known to be NP-hard. The best known approximation algorithm for the EC-SNDP has performance guarantee of 2 (due to Jain [11]), and iteratively rounds solutions to a linear programming relaxation of the problem. ELC has a primal-dual $O(\log k)$-approximation algorithm, where $k = \max_{i,j} r_{ij}$ (Jain et al. [12]). VC-SNDP is not known to have a non-trivial approximation algorithm; however, recently Fleischer [7] has shown how to extend the technique of Jain [11] to give a 2-approximation algorithm in the case that $r_{ij} \in \{0, 1, 2\}$. She also shows that the same techniques will not work for VC-SNDP for more general values of r_{ij}.

In this paper we show that these techniques can be extended to a 2-approximation algorithm for ELC. This gives the first constant approximation algorithm for a general survivable network design problem which allows node failures.

1 Introduction

In the survivable network design problem (SNDP), given an undirected graph and values r_{ij} for each pair of vertices i and j, we attempt to find a minimum-cost subgraph such that there are r_{ij} disjoint paths between vertices i and j. In the edge connected version of this problem (EC-SNDP), these paths must be edge disjoint. In the vertex connected version of the problem (VC-SNDP), the paths must be vertex disjoint. Jain et al. [12] propose a version of the problem intermediate in difficulty to these two, called the *element connectivity problem* (ELC-SNDP, or ELC). In this problem, the set of vertices is partitioned into terminals and nonterminals. The edges and nonterminals of the graph are called *elements*. The values r_{ij} are only specified for pairs of terminals i, j, and the paths from i to j must be element disjoint. Thus if $r_{ij} - 1$ elements fail, terminals i and j are still connected by a path in the network.

The motivation for studying element connectivity is the following: in real networks, both edges (links) and vertices (routers) fail. However, typically network terminals (end hosts) are more robust and located at the fringes of the network. Thus the failure of end hosts is uncommon, and less vital to the connectivity of the network as a whole. Additionally, vertex connectivity problems are much less well understood than edge connectivity problems. Thus, trying to capture node failures by using a vertex connectivity model makes the problem much more difficult. Element connectivity allows the modeling of node failures, while, as we will show in this paper, it shares some of the nice structure that edge connectivity problems have.

The three variants of the survivable network design problem are all NP-hard, since they all include the Steiner tree problem as a special case. Hence we consider *approximation algorithms* for these problems. We say we have a ρ-approximation algorithm for a problem if we have a

polynomial-time algorithm which produces a solution of value no more than ρ times the value of an optimal solution.

The best approximation algorithm for EC-SNDP known is a 2-approximation algorithm due to Jain [11]. This algorithm improved upon a primal-dual $2H_k$-approximation algorithm for EC-SNDP of Goemans et al. [9], where $k = \max_{i,j} r_{ij}$ and $H_k = 1 + \frac{1}{2} + \cdots + \frac{1}{k}$. Jain's algorithm is in fact somewhat more general, and gives an algorithm with performance guarantee 2 for selecting a minimum-cost set of edges such that at least $g(S)$ edges are selected from every cut $\delta(S) = \{(u,v) \in E : u \in S, v \notin S\}$, when g is a weakly supermodular function [9]. The algorithm runs in polynomial time given the existence of a polynomial-time separation algorithm. The EC-SNDP problem corresponds to a particular weakly supermodular function f, and the polynomial-time separation algorithm exists in this case. Jain considers a linear programming relaxation of the problem which has a variable $x(e)$ for each edge e of the graph. The central result of the paper is a theorem showing that any basic solution to the LP will contain a variable $x(e)$ of value at least 1/2. His algorithm builds up a solution by solving the linear programming relaxation, adding to the solution all edges e with $x(e) \geq 1/2$, then iterating on the remaining subproblem. Rounding up each $x(e)$ from 1/2 to 1 gives the performance guarantee of 2 for the algorithm.

No non-trivial approximation algorithm for VC-SNDP is currently known. However, approximation algorithms are known in special cases. In the case $r_{ij} = k$ for all i, j, there is a k-approximation algorithm due to Kortsarz and Nutov [14];[1] furthermore, when edge costs obey the triangle inequality, Khuller and Ragavachari [13] give a constant approximation algorithm. Very recently, Cheriyan, Vempala, and Vetta [3] announced a $6H_k$-approximation algorithm for this problem for graphs that contain at least $6k^2$ vertices. In the case $r_{ij} \in \{0, 1, 2\}$, Ravi and Williamson [15] give a primal-dual 3-approximation algorithm. Recently, Fleischer [7] showed how to extend the algorithm and the proof of Jain for EC-SNDP to this special case of VC-SNDP, obtaining a 2-approximation algorithm.

Our central result is a generalization of Jain's theorem to a new class of functions we call *weakly two-supermodular*. This allows us to give a 2-approximation algorithm for the problem of selecting a minimum-cost set of edges such that there are least $f(S, S')$ edges from $\delta(S, S') = \{(u,v) \in E : u \in S, v \in S'\}$ when f is a weakly two-supermodular function, and a polynomial-time separation algorithm exists. This result specializes to Jain's result exactly in the case that $f(S, S')$ is only non-zero when $S' = V - S$. In this case, $g(S) = f(S, V - S)$ is a weakly supermodular function. The weakly two-supermodular functions are related to bisupermodular functions, which are the negative of bisubmodular functions. Bisubmodular functions appear as increasing rank functions in [17], and in more general form in [8].[2]

As an application of our theorem, we give a 2-approximation algorithm for the element connectivity problem. This improves on a previously known primal-dual $2H_k$-approximation algorithm for ELC due to Jain et al. [12] (a $2H_k$-approximation algorithm for ELC is also obtained as a special case of an algorithm by Zhao, Nagamochi, and Ibaraki [19]). Our algorithm gives the first constant approximation algorithm for a general survivable network design problem which allows node failures. To achieve this result, we introduce a new integer programming formulation for the element connectivity problem, derived from a formulation of VC-SNDP due to Stoer [18].

The connectivity requirement functions for general vertex connectivity are not weakly two-supermodular. Thus our theorem does not apply to these problems. In fact, such a theorem is not possible for general vertex connectivity, as demonstrated in a family of examples developed by the first author in [7]. She shows that there exist a family of vertex connectivity problem instances of where there is a basic solution with $x \leq 1/\sqrt{|E|}$.

In related work, Cheriyan and Vempala [2] have also considered problems of selecting a minimum-cost set of edges from pairs of sets, but in the case of directed graphs. In their problems, one must select $f(S, S')$ edges of those edges directed from a vertex in S to a vertex in S'. They consider crossing bisupermodular functions (a generalization of bisupermodular functions), and show any basic solution to the corresponding LP relaxation contains an edge e such that $x(e) \geq \Omega(1/\sqrt{|E|})$. This model includes uniform vertex connectivity as a special case, but does not include general vertex connectivity. They also show that this is the best possible result for their general model, in the sense that there exists a family of functions f and problem instances for which there is a basic solution with $x \leq O(1/\sqrt{|E|})$.

Our paper is structured as follows. In Section 2, we introduce some notation that we will be using, review Jain's theorem, and state our main theorems. In Section 3, we give the integer programming formulation for ELC, and show that our main theorem gives a 2-approximation algorithm for this problem. Section 4 contains the proof of our central theorem. Section 5 discusses some implementation issues for the 2-approximation algorithm for ELC.

[1] Ravi and Williamson [15] had claimed a $2H_k$-approximation algorithm for the case of general edge costs, but there is an error in their paper; see [16] for details.

[2] The term "bisubmodular" is used with several different meanings in the combinatorial literature. The one we refer to is defined in the context of connectivity problems. A second definition is used in the context of extensions of matching problems [1, 6] and is not equivalent to the first.

2 Preliminaries

Let $G = (V, E)$ be an undirected graph. Let $c(e)$ be a nonnegative cost for each $e \in E$. Let $\delta(S) = \{(u, v) \in E : u \in S, v \notin S\}$. Let $\delta(S, S') = \{(u, v) \in E : u \in S, v \in S'\}$. In this paper, we will only refer to $\delta(S, S')$ when $S \cap S' = \emptyset$. For a set of edges $F \subseteq E$, let $\delta_F(S) = \delta(S) \cap F$ and $\delta_F(S, S') = \delta(S, S') \cap F$. For $x \in \mathbb{R}^{|E|}$, we let $x(S) = \sum_{e \in \delta(S)} x_e$ and $x(S, S') = \sum_{e \in \delta(S, S')} x_e$. Let $g : 2^V \to \mathbb{N}$. Consider the following integer program:

$$\begin{array}{lrcll} \min & \sum_{e \in E} c(e)x(e) & & & \\ \text{s.t.} & x(S) & \geq & g(S), & \forall S \subset V \\ & x(e) & \in & \{0, 1\}, & \forall e \in E. \end{array} \qquad (\mathbf{EC-SNDP})$$

The variable $x(e)$ indicates whether an edge e is in the solution (if $x(e) = 1$) or not. Thus an optimal solution to this integer program finds a minimum-cost set of edges such that there are at least $g(S)$ edges selected from $\delta(S)$ for each $S \subseteq V$. When $g(S) = \max_{i \in S, j \notin S} r_{ij}$, an optimal solution to this integer program gives the solution to the EC-SNDP problem. This is not hard to see, since for any $i, j \in V$, there must be at least r_{ij} edges selected from each cut $\delta(S)$ separating i from j; thus, there are at least r_{ij} edge-disjoint paths from i to j.

We say that g is a *weakly supermodular* function if for any sets $S, T \subseteq V$,

$$\begin{aligned} & g(S) + g(T) \\ & \leq \max\{g(S \cup T) + g(S \cap T), g(S - T) + g(T - S)\}. \end{aligned}$$

It is not hard to prove that the function above for EC-SNDP is weakly supermodular [9].

The main theorem of Jain's paper [11] concerns the linear programming relaxation of the integer program ($\mathbf{EC-SNDP}$) in which the integrality constraints $x(e) \in \{0, 1\}$ are replaced by linear constraints $0 \leq x(e) \leq 1$. The theorem states that any basic solution[3] to this linear program contains some edge e such that $x(e) \geq 1/2$. Even though the linear program contains an exponential number of constraints, a basic, optimal solution to this linear program can be found in polynomial time as long as there exists a polynomial-time *separation oracle* [10]. Given any x, a separation oracle either verifies that x is a feasible solution to the linear program or returns a constraint of the LP violated by x.

Given a polynomial-time separation oracle, the 2-approximation algorithm of [11] works as follows. We start with an empty set of edges $F = \emptyset$. We solve the linear programming relaxation of the problem, and add to F all edges e such that $x(e) \geq 1/2$. We then iterate, now solving the linear programming relaxation with $g'(S) = g(S) - |\delta_F(S)|$. It is not hard to show that if g is weakly supermodular, then so is g'. The algorithm terminates when F is a feasible solution to the problem. Essentially the proof of the performance guarantee compares the cost of the edges added to F in each iteration with its cost in the linear programming relaxation. Since $x(e) \geq 1/2$, its cost is no more than twice its contribution to the linear programming relaxation. Furthermore, since the linear programming relaxation is a lower bound on the cost of an optimal integral solution, this implies that the cost of F is no more than twice optimal. It is easy to give a polynomial-time separation oracle for the function $g(S)$ that defines EC-SNDP (and for functions g' that arise in later iterations); thus this algorithm is a 2-approximation algorithm for EC-SNDP.

To state our central result, we first need a few definitions. The following definitions generalize the one-set function notions of submodularity, supermodularity, and weak supermodularity, and are related to the two-set notions of bisubmodularity and bisupermodularity. A two-set function f defined on the set of pairs of disjoint subsets of V that satisfies

$$\begin{aligned} & f(S, S') + f(T, T') \qquad (1) \\ & \geq \max\{ \quad f(S \cup T, S' \cap T') + f(S \cap T, S' \cup T'), \\ & \qquad f(S \cap T', S' \cup T) + f(S' \cap T, S \cup T') \} \end{aligned}$$

will be called *two-submodular*. A two-set function f is called *bisubmodular* if it obeys only the "first part" of the inequality, namely,

$$f(S, S') + f(T, T') \geq f(S \cup T, S' \cap T') + f(S \cap T, S' \cup T').$$

If we let $S' = V - S$ and $T' = V - T$, then two-submodularity reduces to submodularity for symmetric one-set functions.

If $-f$ is two-submodular, then f is *two-supermodular*. This definition is equivalent to replacing $\geq$ with $\leq$ and $\max$ with $\min$ in the above definition. A two-set function f is *weakly two-supermodular* if

$$\begin{aligned} & f(S, S') + f(T, T') \qquad (2) \\ & \leq \max\{ \quad f(S \cup T, S' \cap T') + f(S \cap T, S' \cup T'), \quad (3) \\ & \qquad f(S \cap T', S' \cup T) + f(S' \cap T, S \cup T') \} \quad (4) \end{aligned}$$

That is, we simply reverse the inequality from our definition of two-submodular functions, without replacing the $\max$ by a $\min$. If we let $S' = V - S$ and $T' = V - T$, weak two-supermodularity reduces to weak supermodularity.

We consider the following linear program, for a function

[3] We refer the reader unfamiliar with the concept of a basic solution of a linear program to a standard textbook on linear programming, such as Chvatal [4]. It is sufficient to say that there exists an optimal solution that is basic, and that it can be found in polynomial time.

$f : 2^{V \times V} \to \mathbb{N}$, and $Q \subseteq V$:

$$\begin{array}{lrcll} \min & \sum_{e \in E} c(e)x(e) & & & \\ \text{s.t.} & x(S,S') & \geq & f(S,S'), & \\ & & & \forall S, S' \subset V, S \cap S' = \emptyset, & \\ & & & V - S - S' \subseteq Q & \\ & 0 \leq x(e) & \leq & 1, \qquad \forall e \in E. & \end{array}$$

(WTS)

We can now state our central theorem, which we prove in Section 4.

Theorem 2.1 *For any weakly two-supermodular function f, any basic solution to (**WTS**) has at least one variable e such that $x(e) \geq 1/2$.*

Given a polynomial-time separation oracle for (**WTS**), we can give a 2-approximation algorithm for solving the integer program associated with (**WTS**). The algorithm is the same as Jain's algorithm given above. First, we find an optimal, basic solution x^* to (**WTS**). Let F be the set of all edges e with $x^*(e) \geq 1/2$. Let $E_{res} = E - F$, and consider the resulting residual LP:

$$\begin{array}{lrcll} \min & \sum_{e \in E_{res}} c(e)x(e) & & & \\ \text{s.t.} & x(S,S') & \geq & f(S,S') - |\delta_F(S,S')|, & \\ & & & \forall\ S, S' \subset V,\ S \cap S' = \emptyset, & \\ & & & V - S - S' \subseteq Q & \\ & 0 \leq x(e) & \leq & 1, \qquad \forall e \in E_{res} & \end{array}$$

(RES)

Let F_{res} be the set of edges returned by recursively applying the algorithm to the residual problem. The following theorem shows that the final set of edges returned by the algorithm is within a factor of 2 of an optimal solution. Let z^*_{res} be the optimal value of the residual LP and z^* be the optimal value of (**WTS**).

Theorem 2.2 *If F_{res} is an integral solution to (**RES**) with value at most $2z^*_{res}$, then $F_{res} \cup F$ is an integral solution to (**WTS**) with value at most $2z^*$.*

Proof: (Sketch.) Follows from same arguments as in [11]. ■

In order to apply the algorithm recursively, we need to show the following lemmas.

Lemma 2.3 *The functions $h(S,S') = |\delta_F(S,S')|$ and $x(S,S')$ are two-submodular.*

Proof: The proofs in both cases follow from a simple counting argument that shows that any edge counted on the right-hand side of (1) also appears on the left-hand side. ■

Lemma 2.4 *If f is weakly two-supermodular and g is two-submodular, then $f - g$ is weakly two-supermodular.*

Proof: Whichever term of the definition of weak two-supermodularity (3) and (4) achieves the maximum, $-g(S,S') - g(T,T')$ satisfies the same inequality, by (1). Hence $(f-g)(S,S') + (f-g)(T,T')$ satisfies this inequality. ■

Corollary 2.5 *If f is weakly two-supermodular, then the function $f(S,S') - |\delta_F(S,S')|$ is also weakly two-supermodular for any set of edges $F \subseteq E$.*

The arguments above yield the following theorem.

Theorem 2.6 *Given a polynomial-time separation oracle for (**WTS**) and (**RES**), the algorithm above is a 2-approximation algorithm for finding the minimum-cost integer solution to (**WTS**).*

In the next section, we show how the element survivable network design problem can be posed as finding an optimal solution to an integer program of the form (**WTS**) with a particular weakly two-supermodular function f. In Section 5 we show that there is a polynomial-time algorithm for finding violated constraints of (**WTS**) and (**RES**) that arise in the execution of the algorithm with this function f. Thus by Theorem 2.6, the algorithm above is a 2-approximation algorithm for ELC.

3 Formulation of ELC

We now turn to an integer programming formulation for the ELC problem, and show that it can be cast in the form of the integer version of (**WTS**) with a weakly two-supermodular function.

We begin by defining some notation. We partition V into R and Q, where R is the set of terminals (or *required* vertices) and Q is the set of nonterminals. This set Q of nonterminals will be the set of vertices Q in the definition of the LP (**WTS**). Given $S \subseteq V$, we define $E(S)$ as the set of edges with both endpoints in S. Given $X \subset E \cup V$, define $G - X$ to be that graph obtained from G by removing all edges and vertices in X, and all edges in $\delta(X \cap V) \cup E(X \cap V)$. We say that X *separates* i from j if i and j are not in the same connected component of $G - X$.

The constraints of our integer program are based on a theorem of Menger (for multiple proofs and references, see [5]).

Theorem 3.1 (Menger) *Let $G = (V,E)$ be a graph, and $s, t \in V$ such that $(s,t) \notin E$. Then, the minimum number of vertices separating s from t in G is equal to the maximum number of vertex disjoint paths from s to t in G.*

Corollary 3.2 *Let $G = (V,E)$ be a graph, with $V = R \cup Q$, $R \cap Q = \emptyset$ and $s, t \in R$. Then, the minimum number*

of elements in $E \cup Q$ separating s from t in G is equal to the maximum number of element disjoint paths from s to t in G. ■

Let r_{ij} denote the element connectivity requirements between any pair of terminals $i, j \in R$. Our corollary to Menger's theorem says that G satisfies the element connectivity requirements if for every subset $X \subseteq E \cup Q - \{i, j\}$ with $|X| < r_{ij}$, i and j are in the same connected component of $G - X$. For any two disjoint subsets $S, S' \subset V$, with all remaining vertices being nonterminals (that is, $V - S - S' \subseteq Q$), we define the two-set function f_r by $f_r(S, S') := \max\{r_{ij} | i \in S \cap R, j \in S' \cap R\}$. We assume that $f_r(S, S')$ is undefined if S and S' are not disjoint or if $V - S - S' \not\subseteq Q$. We further assume that $r_{ij} \in \mathbb{Z}$ for all $i, j \in R$, and that $f_r(S, S') = 0$ if either S or S' is the empty set and is otherwise well-defined.

Since there may be at most one element-disjoint path from S to S' through each vertex in $V - S - S'$, in order to have a feasible solution to the element connectivity problem, the number of edges from S to S' must therefore be at least $f_r(S, S') - |V - S - S'|$. Thus we define the two-set function g_r by $g_r(S, S') = f_r(S, S') - |V - S - S'|$. The following lemma is a simple consequence of Corollary 3.2.

Lemma 3.3 *The set of integral solutions to the LP (**WTS**) with the function $f = g_r$ equals the set of solutions to the corresponding element connectivity problem.* ■

In order to show that Theorem 2.1 applies to this linear programming formulation, we need to show that g_r is a weakly two-supermodular function. We begin with some notation. Given (S, S'), there is a pair $i \in S \cap R$, $j \in S' \cap R$ that determines $f_r(S, S')$. Let $i(S, S')$ denote one such i and $j(S, S')$ denote the corresponding j.

Lemma 3.4 *The two-set function f_r is weakly two-supermodular.*

Proof: Before we can start the proof, we need to argue that the function value f_r is well-defined on the arguments of f_r used in the definition of weakly two-supermodular. We assume that S and S' are disjoint, that T and T' are disjoint, that $V - S - S' \subseteq Q$, and that $V - T - T' \subseteq Q$. We need to show (for example) that $f_r(S \cap T, S' \cup T')$ is well-defined. First, $S \cap T$ and $S' \cup T'$ are disjoint because any element in S is not in S' and any element in T is not in T', so that any element in $S \cap T$ cannot be in $S' \cup T'$. Second, we need to show that any element not in $(S \cap T) \cup (S' \cup T')$ must be in Q. This follows since any element not in these two sets must be in either $V - S - T - S' - T'$ (and thus is certainly in Q) or $S - T - T'$ (and thus in $V - T - T'$, which implies membership in Q) or $T - S - S'$ (and thus in $V - S - S'$, which implies membership in Q). The other cases are similar.

We now show that f_r is weakly two-supermodular. For any pair (T, T'), note that since $V - T - T'$ contains only nonterminals, $i(S, S') \in (S \cap T) \cup (S \cap T')$ and $j(S, S') \in (S' \cap T) \cup (S' \cap T')$.

Call sets $S \cap T$ and $S' \cap T'$ complements, and sets $S \cap T'$ and $S' \cap T$ complements. The set $I = \{i(S, S'), i(T, T'), j(S, S'), j(T, T')\}$ intersects two, three, or all four of these sets. If only two, then these two sets are complements, $f(S, S') = f(T, T')$, and either (4) or (3) holds, depending on the set of complements. If I intersects three sets, then two of these are complements with the property that a requirement vertex for (S, S') is contained in at least one set of the complementary pair, and a requirement vertex for (T, T') is contained in the complementary set. Thus the inequality of (4) or (3) corresponding to this complementary pair holds. Finally, we have the case when I intersects all four sets. In this case, select the complementary pair that contains a requirement vertex of pair (i, j) with $r_{ij} = \max\{f(S, S'), f(T, T')\}$. Then the inequality that corresponds to this complementary pair holds. ■

Lemma 3.5 *The two-set function $g_r(S, S')$ is weakly two-supermodular.*

Proof: The function $h(S, S') := |V - S - S'|$ is two-submodular, and satisfies (1) with equality. Thus by Lemma 2.4, $f_r(S, S') - |V - S - S'|$ is weakly two-supermodular. ■

4 Proof of the main theorem

We now turn to the proof of our main theorem, Theorem 2.1. Our basic proof outline follows that of Jain [11]. Before we can sketch the proof outline, we need to define some terms. We say that a pair of sets S, T *cross* if all three of $S \cap T$, $S - T$, and $T - S$ are nonempty. A collection of sets is *laminar* if no pair of sets in the collection cross. Given a feasible solution x to the LP (**EC − SNDP**), we say that the constraint corresponding to the set $S \subset V$ is *tight* if $x(S) = f(S)$. The proof in [11] shows first if a pair of tight sets S and T cross, then either $S \cup T$ and $S \cap T$ are tight, or $S - T$ and $T - S$ are tight. Furthermore, there is a linear relationship between the edges with exactly one endpoint in these sets. This "uncrossing lemma" is used to prove that there exists a basic solution corresponding to a laminar collection of tight sets. This laminar collection defines a partial order on the sets of the collection via the subset relation. This poset has a forest structure, and the forest is used to prove the existence of an edge of value at least 1/2.

Here we prove analogues of the uncrossing lemma (Lemma 4.1) and the laminar basis lemma (Corollary 4.4).

We then define an analogous poset of a laminar collection which has a forest structure (Lemmas 4.6, 4.7, 4.8). Given the forest, we invoke a lemma from [11] to prove the existence of the edge e with $x(e) \geq 1/2$. The central technical difficulty lies in deriving the appropriate analogs of the concepts of "cross", "laminar", and the poset when dealing with pairs of sets; and in defining the appropriate analog of "incidence" so that a charging argument similar to that in [11] works. Additionally, proofs that are quite simple in the single set world become non-trivial in the pairs of sets world (e.g. Lemma 4.2).

We now begin the proof. Let x be a feasible solution to (**WTS**) for a weakly two-supermodular function f. A set pair (S, S') is *tight* if $x(S, S') = f(S, S')$. In particular, we will be interested in the case when x is a basic solution to (**WTS**) with the property that $x(e) < 1$ for all $e \in E$ (since otherwise Theorem 2.1 holds trivially). In this case, we define E_x to be the set of edges with nonzero x-value. Given x, define $\chi_x(S, S') \in \{0, 1\}^{|E|}$ to be the characteristic vector of the support of $x(S, S')$.

Lemma 4.1 (Uncrossing Lemma) *If (S, S') and (T, T') are tight for a weakly two supermodular function f with respect to x, then one of the following holds.*

i. *$(S \cap T, S' \cup T')$ and $(S \cup T, S' \cap T')$ are tight, and $\chi_x(S, S') + \chi_x(T, T') = \chi_x(S \cup T, S' \cap T') + \chi_x(S \cap T, S' \cup T')$,*

ii. *$(S \cap T', S' \cup T)$ and $(S' \cap T, S \cup T')$ are tight and $\chi_x(S, S') + \chi_x(T, T') = \chi_x(S \cap T', S' \cup T) + \chi_x(S' \cap T, S \cup T')$.*

Proof: Because f is weakly two-supermodular, either (3) or (4) holds for (S, S') and (T, T'); suppose (3) holds, and $f(S, S') + f(T, T') \leq f(S \cup T, S' \cap T') + f(S \cap T, S' \cup T')$ (the other case is similar). We know that $x(S, S')$ is two-submodular by Lemma 2.3. Let $h(S, S') = f(S, S') - x(S, S')$. Thus $h(S, S') + h(T, T') \leq h(S \cup T, S' \cap T') + h(S \cap T, S' \cup T')$. By the feasibility of x, $x(X, X') \geq f(X, X')$, so that $h(X, X') \leq 0$ for all set pairs (X, X'). For (S, S') and (T, T') that are tight for f with respect to x, $h(S, S') = h(T, T') = 0$. Thus we know $h(S, S') + h(T, T') = 0$, $h(S \cup T, S' \cap T') \leq 0$, and $h(S \cap T, S' \cup T') \leq 0$, so it must be the case that $h(S \cup T, S' \cap T') = h(S \cap T, S' \cup T') = 0$, and $(S \cup T, S' \cap T')$ and $(S \cap T, S' \cup T')$ are tight. It is not hard to show that $\chi_x(S, S') + \chi_x(T, T') \geq \chi_x(S \cup T, S' \cap T') + \chi_x(S \cap T, S' \cap T')$ by showing that every edge that appears in the right-hand side of the inequality must appear in the left-hand side. We need to show that equality holds; suppose not, and suppose the inequality is strict. It then follows that $x(S, S') + x(T, T') > x(S \cup T, S' \cap T') + x(S \cap T, S' \cap T')$. But then $0 = h(S, S') + h(T, T') < h(S \cup T, S' \cap T') + h(S \cap T, S' \cup T') = 0$, a contradiction. ■

We define a relation $\leq$ on set pairs by $(S, S') \leq (T, T')$ if $S \subseteq T$ and $S' \supseteq T'$. It is easy to check that this relation is transitive, reflexive, and anti-symmetric, and hence defines a partial order. We say that the pairs (S, S') and (T, T') *pair-cross* if they do not satisfy either of the following two conditions: (1) (S, S') and (T, T') are comparable in the partial order (that is, either $(S, S') \leq (T, T')$ or vice versa); (2) $S \subseteq T'$ and $T \subseteq S'$. A collection $\mathcal{L}$ of pairs (S, S') is called *pair-laminar* if no two pairs in $\mathcal{L}$ pair-cross.

It is important to note that this definition is not symmetric: It is possible that (S, S') and (T, T') do not pair-cross while (S', S) and (T', T) do pair-cross. In this case, the uncrossing lemma applied to (S', S) and (T', T) will return (S, S') and (T, T'). This asymmetry is necessary in the following sense: if we use a perhaps more natural definition of crossing that defines two set pairs to cross if any two of their defining sets cross, then it may be the case that we have crossing set pairs, but that after applying the uncrossing lemma, we remain with the same two set pairs. For instance, consider the case when $T \subset S$, but S and T' cross. This happens when $S \not\subseteq T \cup T'$. In this case, condition (ii) of the uncrossing lemma does not hold, as $S' \cap T = \emptyset$; but condition (i) of the uncrossing lemma does hold. However, $S \cup T = S$ and $S \cap T = T$, so that the same two set pairs are returned after applying the uncrossing lemma. Thus we must relax this definition so that every application of the uncrossing lemma yields two set pairs which do not pair-cross. We cannot relax it too much, however, since we require specific properties of pair-laminar in order to obtain our result. In particular, a pair-laminar collection of set pairs must define a poset which has a forest structure. We prove that this is the case in Lemma 4.6. The second condition of pair-cross is also used critically in Lemma 4.7.

The following technical lemma is central to the validity of our main theorem. The proof contains many cases because the definition of pair-cross is not symmetric.

Lemma 4.2 *Let (S, S') and (T, T') be pairs that pair-cross. If (X, X') does not pair-cross either (S, S') or (T, T'), then it does not pair-cross any of the pairs:*
a*: $(S \cap T, S' \cup T')$,*
b*: $(S \cup T, S' \cap T')$,*
c*: $(S \cap T', S' \cup T)$,*
d*: $(S' \cap T, S \cup T')$.*

Proof: We note that if $(S, S') \leq (X, X')$ then it follows immediately that $(S \cap T, S' \cup T') \leq (S, S') \leq (X, X')$ and that $(S \cap T', S' \cup T) \leq (S, S') \leq (X, X')$; that is, (X, X') does not pair-cross **a** or **c**. Similarly, if $(S, S') \geq (X, X')$ then $(S \cup T, S' \cap T') \geq (S, S') \geq (X, X')$, and (X, X') does not pair-cross $(S' \cap T, S \cup T')$ since $S' \cap T \subseteq S' \subseteq X'$ and $X \subseteq S \subseteq S \cup T'$, and thus (X, X') does not pair-cross **b** or **d**. Similarly, if $(T, T') \leq (X, X')$ then $(S \cap T, S' \cup T') \leq (X, X')$ and $(S' \cap T, S \cup T') \leq (X, X')$, so that

(X, X') does not pair-cross **a** or **d**. If $(T, T') \geq (X, X')$ then $(S \cup T, S' \cap T') \geq (X, X')$ and (X, X') does not pair-cross $(S \cap T', S' \cup T)$ since $X \subseteq S' \cup T$ and $S \cap T' \subseteq X'$, so that (X, X') does not pair-cross **b** or **c**.

We now exhaustively enumerate cases. If both $(S, S'), (T, T') \leq (X, X')$, then by the above (X, X') does not pair-cross **a**, **c**, or **d**. Furthermore, it does not pair-cross **b** since we know $S, T \subseteq X$ and $S', T' \supseteq X'$, and therefore $(S \cup T, S' \cap T') \leq (X, X')$.

If both $(S, S'), (T, T') \geq (X, X')$, then by the above (X, X') does not pair-cross **b**, **c**, or **d**. It does not pair-cross **a** since we know $S, T \supseteq X$ and $S', T' \subseteq X'$ and therefore $(S \cap T, S' \cup T') \geq (X, X')$.

The case $(S, S') \leq (X, X') \leq (T, T')$ or the reverse cannot occur because this implies that (S, S') and (T, T') do not pair-cross, contradicting the hypothesis.

Note that because (S, S') and (T, T') pair-cross, it cannot be the case that $(S, S') \leq (X, X')$ and $X \subseteq T'$ and $T \subseteq X'$ since this implies that $S \subseteq T'$ and $T \subseteq S'$. Similarly, this case with (S, S') interchanged with (T, T') cannot occur. So we assume $(S, S') \geq (X, X')$ and that $X \subseteq T'$ and $T \subseteq X'$. From the above, we know that (X, X') does not pair-cross **b** or **d**. Furthermore, it does not pair-cross **a** since $X \subseteq S' \cup T'$ and $S \cap T \subseteq X'$. It does not pair-cross **c** since $(X, X') \leq (S \cap T', S' \cup T)$. The case with (S, S') and (T, T') interchanged is similar.

Finally, we assume that (X, X') does not pair-cross (S, S') because $X \subseteq S'$ and $S \subseteq X'$, and does not pair-cross (T, T') because $X \subseteq T'$ and $T \subseteq X'$. It follows easily that (X, X') does not pair-cross **a** because $X \subseteq S' \cup T'$ and $S \cap T \subseteq X'$. It does not pair-cross **b** because $X \subseteq S' \cap T'$ and $S \cup T \subseteq X'$. It does not pair-cross **c** because $X \subseteq S' \cup T$ and $S \cap T' \subseteq X'$, and it does not pair-cross **d** because $X \subseteq S \cup T'$ and $S' \cap T \subseteq X'$. ■

Lemma 4.3 *Given pairs (S, S') and (T, T'), neither (S, S') nor (T, T') pair-cross any of the four pairs $(S \cap T, S' \cup T')$, $(S \cup T, S' \cap T')$, $(S \cap T', S' \cup T)$, $(S' \cap T, S \cup T')$.*

Proof: Consider (S, S'); the proof for (T, T') is analogous. We see that $(S \cap T, S' \cup T') \leq (S, S')$, $(S, S') \leq (S \cup T, S' \cap T')$, $(S \cap T', S' \cup T) \leq (S, S')$, and $S \subseteq S \cup T'$ and $S' \cap T \subseteq S'$. Thus (S, S') does not cross any of these four sets. ■

Let $\mathcal{T}$ be the collection of all tight set pairs with respect to x. Given a collection of set pairs $\mathcal{A}$, we define $\mathrm{Span}(\mathcal{A})$ to be the vector space spanned by the characteristic vectors $\chi_x(S, S')$ of the set pairs $(S, S') \in \mathcal{A}$.

Lemma 4.4 *For any maximal pair-laminar collection $\mathcal{L}$ of tight set pairs, $\mathrm{Span}(\mathcal{L}) = \mathrm{Span}(\mathcal{T})$.*

Proof: If $\mathrm{Span}(\mathcal{L}) \neq \mathrm{Span}(\mathcal{T})$, then $\mathrm{Span}(\mathcal{L}) \subset \mathrm{Span}(\mathcal{T})$ since $\mathcal{L} \subset \mathcal{T}$. Thus there exists a pair $(S, S') \in \mathcal{T}$, with $\chi_x(S, S') \notin \mathrm{Span}(\mathcal{L})$, such that (S, S') pair-crosses a minimum number of set pairs in $\mathcal{L}$. Let (T, T') be one of those pairs. Then by Lemma 4.1, we can rewrite $\chi_x(S, S')$ as a linear combination of characteristic vectors of pair-laminar tight set pairs. Suppose that condition (i) of Lemma 4.1 holds; thus we can rewrite $\chi_x(S, S')$ as $\chi_x(S \cup T, S' \cap T') + \chi_x(S \cap T, S' \cup T') - \chi_x(T, T')$ (the case in which condition (ii) holds is similar). Since $\chi_x(S, S') \notin \mathrm{Span}(\mathcal{L})$, at least one of $\chi_x(S \cup T, S' \cap T')$ and $\chi_x(S \cap T, S' \cup T')$ is also not in $\mathrm{Span}(\mathcal{L})$. By Lemma 4.2, any set pair $(X, X') \in \mathcal{L}$ pair-crossing either of $(S \cup T, S' \cap T')$ and $(S \cap T, S' \cup T')$ must also have pair-crossed (S, S'), since (X, X') does not pair-cross (T, T') by the laminarity of $\mathcal{L}$. Furthermore, by Lemma 4.3 the set pairs $(S \cup T, S' \cap T')$ and $(S \cap T, S' \cup T')$ do not pair-cross (T, T'). Since these set pairs do not pair-cross (T, T'), they have strictly fewer crossings with sets in $\mathcal{L}$ than (S, S') does, contradicting the choice of (S, S'). ■

The corollary below follows from Lemma 4.4 and the properties of a basic solution.

Corollary 4.5 *There exists a pair-laminar set $\mathcal{B}$ of tight set pairs satisfying*

1. $|\mathcal{B}| = |E_x|$,
2. *the vectors $\chi_x(S, S')$ for $(S, S') \in \mathcal{B}$ are linearly independent,*
3. $f(S, S') \geq 1$ *for all* $(S, S') \in \mathcal{B}$. ■

Lemma 4.6 *If $\mathcal{B}$ is a collection of pair-laminar set pairs, then the poset defined by $\leq$ on the set pairs in $\mathcal{B}$ is described by a unique forest.*

Proof: It suffices to establish that if $(S, S') \leq (X, X')$ and $(S, S') \leq (Y, Y')$ and $|X| < |Y|$ or $|X| = |Y|$ and $|X'| \geq |Y'|$ for (S, S'), (X, X'), $(Y, Y') \in \mathcal{B}$ then $(X, X') \leq (Y, Y')$. Since $\mathcal{B}$ is pair-laminar, (X, X') and (Y, Y') do not pair-cross. By the conditions above, it cannot be the case that $(X, X') \geq (Y, Y')$, so either $(X, X') \leq (Y, Y')$ or $X \subseteq Y'$ and $Y \subseteq X'$. However, in the latter case we know $S \subseteq X \subseteq Y'$, and since Y and Y' are disjoint, this implies $S \not\subseteq Y$, contradicting $(S, S') \leq (Y, Y')$. So it must be the case that $(X, X') \leq (Y, Y')$. ■

To prove Theorem 2.1, we use a proof by contradiction. We use Lemma 4.6 to construct a forest of tight set pairs in $\mathcal{B}$. We then define a new concept of incidence of edges with fractional value to nodes in this forest. Given that no edge has value at least $\frac{1}{2}$, we can then charge edges with fractional value to the nodes in this forest in a way that leads to a contradiction.

Start with a pair-laminar family $\mathcal{B}$ as given by Corollary 4.5. Form the rooted forest corresponding to the containment poset on $\mathcal{B}$ as indicated in Lemma 4.6: the node set is $\mathcal{B}$ and there is an arc from $(S, S') \in \mathcal{B}$ to $(T, T') \in \mathcal{B}$

if (T,T') is the smallest pair of the partial order such that $(S,S') \neq (T,T')$ and $(S,S') \leq (T,T')$. Note that we refer to *nodes* and *arcs* of the forest, while we use *vertices* and *edges* when refering to the original graph.

A *socket* (e,i) is a pairing of an edge $e \in E_x$ with one of its two endpoints. Each edge $e=(i,j) \in E_x$ is associated with two sockets: (e,i) and (e,j). We assign each socket to at most one node of the forest; we say that the socket is *incident* to that node. For an edge $e=(i,j) \in E_x$, the socket (e,i) is incident to node (S,S') if (S,S') is the lowest node in the tree among all nodes with either $i \in S$ or $\{i,j\} \cap S' = \emptyset$.

Lemma 4.7 *Incidence is well-defined.*

Proof: We need to show that for a given socket (e,i), the definition of incidence is such that the socket is assigned to at most one node. It suffices to show that any two set pairs $(S,S'),(T,T') \in \mathcal{B}$ for which one of the two conditions of incidence holds must be comparable in the partial order.

First, if $i \in S$ and $i \in T$ then since (S,S') and (T,T') don't pair-cross, it must be the case that either $(S,S') \leq (T,T')$ or $(T,T') \leq (S,S')$.

Now suppose that $i \in S$, $i \notin T$, and $\{i,j\} \cap T' = \emptyset$. We know that (S,S') and (T,T') do not pair-cross. Since $i \in S$ and $i \notin T$, we know $(S,S') \not\leq (T,T')$. Since $i \in S$ and $i \notin T'$, it is not the case that $S \subseteq T'$ and $T \subseteq S'$. Thus it must be the case that $(T,T') \leq (S,S')$. ■

We say that an edge *crosses* a node (S,S') if exactly one of its associated sockets is incident to any node in the subtree rooted at (S,S').

Lemma 4.8 *An edge (i,j) with fractional value crosses a node (S,S') if and only if $|\{i,j\} \cap S| = |\{i,j\} \cap S'| = 1$.*

Proof: ($\Rightarrow$): For edge $e=(i,j)$ crossing node (S,S'), assume that the socket (e,i) is incident to a node in the subtree rooted at (S,S') and (e,j) is not. Then $i \notin S'$ and $j \in S'$ by definition of incidence and the properties of the partial order. Since $j \in S'$, this in turn implies $i \in S$. ($\Leftarrow$): If $i \in S$ and $j \in S'$ and $\mathcal{B}$ is pair-laminar, then by Lemma 4.7, (e,i) is incident to a node (T,T') satisfying $(T,T') \leq (S,S')$. By Lemma 4.6, this implies that (e,i) is incident to a node in the subtree rooted at (S,S'). Since $j \in S'$, (e,j) is not incident to any $(T,T') \leq (S,S')$. ■

Proof of Theorem 2.1: The theorem is proved by contradiction: suppose every edge takes value strictly less than $\frac{1}{2}$. We show that given this assumption, we can "charge" the sockets incident to any rooted subtree of the forest in such a way that each node gets charged at least two sockets and the root gets charged at least 3 sockets. This leads to a contradiction, since the number of sockets is twice the number of edges, and the number of edges equals the number of nodes in the forest.

This charging scheme is carried out inductively bottom up on the structure of the tree. Consider first a leaf element (S,S') of the forest. Since we know $f(S,S') \geq 1$ and each edge has value less than 1/2, it must be the case that $|\delta_{E_x}(S,S')| \geq 3$. By Lemma 4.8 it must be the case that each of these edges cross (S,S'), which implies that at least three sockets are incident on (S,S'). We invoke a lemma of Jain below in order to carry out the induction.

Let $y_e = \frac{1}{2} - x_e$. Note that by hypothesis $y_e > 0$. For any pair $(S,S') \in \mathcal{B}$, define its *co-requirement* as the sum of y_e's for all the edges crossing (S,S'). The co-requirement satisfies $y(S,S') = \frac{1}{2}|\delta_{E_x}(S,S')| - f(S,S')$. Since $f(S,S')$ is integral, the co-requirement of (S,S') is an integral multiple of $\frac{1}{2}$.

Lemma 4.9 (Jain [11], Lemma 4.6) *For any rooted subtree of the forest, we can charge the sockets incident to it such that every node gets charged at least 2 sockets and the root gets charged at least 3. Moreover, the root gets charged exactly 3 sockets only if its co-requirement is half.*

Note that the statement of the lemma in [11] refers to endpoints in a way that is analogous to our use of the term sockets here. The proof of this lemma for our problem is exactly the same proof in [11] which consists of checking a series of cases. We omit it here due to the similarity to, and the length of, the original. ■

There is an example in [11] that shows that the analysis there is tight for the edge connectivity problem with connectivity requirements in $\{0,1\}$. Since in this case the edge and element connectivity problems are the same, the same example shows that the analysis is also tight for the element connectivity problem.

5 Implementation Issues

To solve the LP in polynomial time, we need a separation oracle for the connectivity constraints: an algorithm that finds a violated constraint of the LP (**WTS**) with the function $g_r(S,S')$ or (**RES**) with the function $g_r(S,S') - |\delta_F(S,S')|$. To do this, we interpret x-values as capacities and transform the graph induced by the current fractional solution x and the fixed edges F into a directed graph by replacing every edge by oppositely oriented edges with the same capacity as the original undirected edge. We then perform a standard procedure of splitting nonterminal vertices to model the fact that at most one path can pass through any nonterminal. Then, in the resulting graph, the maximum flow value between i and j is vertex connectivity between i and j. If this is less than r_{ij}, the minimum cut reveals a violated inequality.

Thus we have a polynomial-time separation oracle for (**WTS**) and (**RES**). Using ellipsoid algorithm, we can then obtain a basic solution in polynomial time [10].

Acknowledgements

The first author acknowledges the partial support provided by NSF through grants EIA-0049084 and CCR-0049071.

References

[1] A. Bouchet and W. Cunningham. Delta-matroids, jump systems and bisubmodular polyhedra. *SIAM J. Discrete Math.*, 8:17-32, 1995.

[2] J. Cheriyan and S. Vempala. Edge covers of setpairs and the iterative rounding method. In *Integer Programming and Combinatorial Optimization*, number 2081 in the Lecture Notes in Computer Science, pages 30–44. Springer, 2001.

[3] J. Cheriyan, S. Vempala, and A. Vetta. An approximation algorithm for the minimum-cost k-vertex connected subgraph. Manuscript, July 2001.

[4] V. Chvatal. *Linear programming*. W. H. Freeman, New York, 1983.

[5] R. Diestel. *Graph Theory*. Number 173 in Graduate Texts in Mathematics. Springer-Verlag, New York, 2nd edition, 2000.

[6] F. D. J. Dunstan and D. J. A. Welsh. A greedy algorithm solving a certain class of linear programmes. *Math. Programming*, 5:338-353, 1973.

[7] L. Fleischer. A 2-approximation for minimum cost $\{0, 1, 2\}$ vertex connectivity. In *Integer Programming and Combinatorial Optimization*, number 2081 in the Lecture Notes in Computer Science, pages 115–129. Springer, 2001.

[8] A. Frank and T. Jordan. Minimal edge-coverings of pairs of sets. *J. Combinatorial Theory B*, 65:73-110, 1995.

[9] 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.

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

[11] K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 21:39–60, 2001.

[12] K. Jain, I. Mandoiu, V. V. Vazirani, and D. P. Williamson. A primal-dual schema based approximation algorithm for the element connectivity problem. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 484–489, 1999.

[13] S. Khuller and B. Raghavachari. Improved approximation algorithms for uniform connectivity problems. *J. Algorithms*, 21:434–450, 1997.

[14] G. Kortsarz and Z. Nutov. Approximating node connectivity problems via set covers. In *Approximation Algorithms for Combinatorial Optimization (Proc. of APPROX 2000)*, number 1913 in Lecture Notes in Comp. Sci., pages 194–205. Springer-Verlag, 2000.

[15] R. Ravi and D. P. Williamson. An approximation algorithm for minimum-cost vertex-connectivity problems. *Algorithmica*, 18(1):21–43, 1997.

[16] R. Ravi and D. P. Williamson. Erratum: An approximation algorithm for minimum-cost vertex-connectivity problems. Manuscript, July 2001. Available at `www.almaden.ibm.com/cs/people/dpw`.

[17] A. Schrijver. Matroids and linking systems. *Journal of Combinatorial Theory B*, 26:349–369, 1979.

[18] M. Stoer. *Design of survivable networks*. Number 1531 in Lecture Notes in Mathematics. Springer-Verlag, Berlin, 1992.

[19] L. Zhao, H. Nagamochi, and T. Ibaraki. A primal-dual approximation algorithm for the survivable network design problem in hypergraph. In *STACS 2001*, number 2010 in Lecture Notes for Computer Science, pages 478–489. Springer-Verlag, 2001.

This research was sponsored in part by National Science Foundation (NSF) grant no. CCR-0122581.