Approximation Algorithms for Minimum-Cost k-Vertex Connected Subgraphs

Joseph Cheriyan*, Santosh Vempala† and Adrian Vetta‡

Abstract. We present two new algorithms for the problem of finding a minimum-cost k-vertex connected spanning subgraph. The first algorithm works on undirected graphs with at least $6k^2$ vertices and achieves an approximation factor of 6 times the kth harmonic number, which is $O(\log k)$. The second algorithm works on directed and undirected graphs. It gives an $O(\sqrt{n/\epsilon})$-approximation algorithm for any $\epsilon > 0$ and $k \leq (1-\epsilon)n$. These algorithms improve on the previous best approximation factor (which was more than $k/2$). The latter algorithm also extends to other problems in network design with vertex connectivity requirements. Our main tools are setpair relaxations, a theorem of Mader's (in the undirected case) and iterative rounding (general case).

1. Introduction

A typical problem in network design is to find a minimum-cost sub-network H of a given network G such that H satisfies some prespecified connectivity requirements; examples are the *minimum spanning tree* (MST) problem and the *traveling salesman problem* (TSP). By a *network* we mean a graph (either undirected or directed) together with non-negative costs for the edges. Problems in network design have a central position in combinatorial optimization and theoretical computer science, and moreover, they arise in many practical settings (e.g., telephone networks). Many of the problems in network design are NP-hard. Over the past decade, there has been significant research on approximation algorithms for network design, and there have been some notable successes in the design of networks that satisfy various types of "edge connectivity" requirements, e.g., Goemans and Williamson [11], Jain [13], etc. Fewer results are known on the design of networks subject to "vertex connectivity" requirements. We will focus on the classic problem in this class, namely the *minimum cost k-vertex connected spanning subgraph (k-VCSS)*. In this problem, we are given a graph $G = (V, E)$ (either directed or undirected) with non-negative edge costs $c : E \to \mathbb{R}_+$, and the task is to find a minimum-cost spanning subgraph H such that H is k-vertex connected. A graph is called *k-vertex connected* if it has at least $k+1$ vertices, and the deletion of any set of $k-1$ vertices leaves a connected graph; equivalently there are at least k vertex-disjoint paths in H between every pair of vertices. The problem is NP-hard for $k \geq 2$; for $k = 1$ it is the minimum spanning tree problem. The analogous problem for edge connectivity is also NP-hard, and a 2-approximation algorithm was found almost a decade ago [16]. Despite much effort, no approximation guarantee better than $\Omega(k)$ was known for the vertex connectivity problem. The best previous approximation guarantee was more than $k/2$ due to Kortsarz and Nutov [17]. Earlier Ravi and Williamson [21] had claimed an $O(\log k)$ approximation guarantee, but subsequently an error was found and that claim has been withdrawn. There are several recent results for restricted edge costs, such as uniform costs or metric costs, but in this paper we discuss only the general case of non-negative costs.

Our results are based on the linear programming relaxation obtained from an integer formulation called the *setpair formulation* (due to Frank and Jordan [5]). Most of the previous research in this area is based upon an integer programming formulation called the *cut covering* model. There a non-negative, integer requirement $f(S, V \setminus S)$ is assigned to each vertex partition $(S, V \setminus S)$,

---

*Dept. of Comb. and Opt., Univ. Waterloo, Waterloo, ON, Canada. Email: `jcheriyan@math.uwaterloo.ca`
†Department of Mathematics, M.I.T., Cambridge. Email: `vempala@math.mit.edu`
‡Department of Mathematics, M.I.T., Cambridge. Email: `avetta@math.mit.edu`

and the goal is to find a minimum-cost subgraph H such that each cut $(S, V \setminus S)$ has at least $f(S, V \setminus S)$ edges of H. (For example, in the MST problem, $f(S, V \setminus S) = 1$ for each nonempty vertex set S, $S \neq V$.) Although this model captures many "edge connectivity" problems in network design, it seems less suited for "vertex connectivity" problems. The setpair formulation [5] is a generalization of the cut covering model. A setpair (W_t, W_h) consists of two disjoint vertex sets W_t and W_h, i.e., $W_t \subseteq V$, $W_h \subseteq V$, $W_t \cap W_h = \emptyset$. Each setpair (W_t, W_h) is assigned an non-negative, integer requirement $f(W_t, W_h)$. The goal is to find a minimum-cost subgraph H that satisfies the requirement of every setpair, i.e., for each setpair (W_t, W_h), H should have at least $f(W_t, W_h)$ edges that have one end-vertex in W_t and the other end-vertex in W_h. Note that the cut covering model is a special case of the setpair formulation in which every setpair (W_t, W_h) with a positive requirement has $W_t = V \setminus W_h$ (i.e., a pair of complementary sets).

Our first algorithm works on undirected graphs with at least $6k^2$ vertices and finds a solution of cost at most 6 times the kth harmonic number, which is $O(\log k)$. It follows that this is an upper bound on the integrality ratio for the linear programming relaxation. The previous best upper bound on the integrality ratio was $\Theta(k)$. This algorithm is based on two results: (1) a polynomial-time algorithm of Frank and Tardos [6] for finding a minimum-cost k-outconnected subdigraph of a digraph (directed graph), and (2) an upper bound on the order of 3-critical graphs by Mader [18]. The Frank-Tardos algorithm has been applied earlier to the mincost k-VCSS problem by several authors, starting with Khuller and Raghavachari [15]; see also [1, 2, 17].

Our second algorithm works for both undirected and directed graphs; it is based on a structural characterization of basic solutions of the relaxation and *iterative rounding.* We introduce a special class of requirement functions f, namely, *skew bisupermodular* functions (see Section 4). These functions are a common generalization of weakly supermodular (symmetric) functions ([4], [10]) and crossing bisupermodular functions [5]. Skew bisupermodular requirement functions are useful because the extreme points of the LP relaxation are characterized by "non-crossing families" of setpairs (see Section 4.4 and Theorem 4.3).

Note the correspondence with the cut covering model, where the extreme points of the LP relaxation may be characterized by a laminar family of sets. Based on this, we obtain lower bounds on the maximum weight of an edge for the basic solutions of the LP relaxation.

The iterative rounding method works as follows. Formulate the problem as an integer program, and then solve the LP (linear programming) relaxation to find a basic (extreme point) optimum solution $\boldsymbol{x}$. Pick an edge e^* of highest weight (i.e., $x_{e^*} \geq x_e$, $\forall e \in E$) and add it to the solution subgraph H (initially, $E(H)$ is empty). Then update the LP and the integer program, by fixing the variable x_{e^*} at value 1, and iteratively solve the reduced problem. The resulting LP is the same as the LP for the "reduced" problem where the edge e^* is pre-selected for H. Under appropriate conditions on the requirement function f, the problem turns out to be "self reducible," i.e., the essential properties of the original problem are preserved in the reduced problem. Jain [13] applied this method to the cut covering model, and proved that it achieves an approximation guarantee of 2 provided that the requirement function f is weakly supermodular. Our relaxations also have the self-reducibility property needed for the iterative rounding method. We obtain an $O(\sqrt{n/\epsilon})$-approximation algorithm for any $\epsilon > 0$ and $k \leq (1-\epsilon)n$. We remark that, to the best of our knowledge, iterative rounding was first applied to the setpairs formulation by the first two authors in [3]. Subsequently, Fleischer et al [7] used it on the element connectivity problem. This paper uses the results of [3] but is independent of [7].

The rest of this paper is organized as follows. Section 2 describes the setpair relaxations. Section 3 gives the first algorithm for k-connectivity based on Mader's theorem. Section 4 contains our main result on skew bisupermodular functions, a combinatorial characterization of the extreme points of the LP and Section 5 gives an approximation algorithm based on iterative rounding. Throughout, we assume that the input graph $G = (V, E)$ is k-vertex connected and n denotes $|V|$.

2. Setpair Formulations and Relaxations

A *setpair* $W = (W_t, W_h)$ is an ordered pair of disjoint vertex sets; either W_t or W_h may be empty. We say that W_t is the *tail* of W, and W_h is the *head* of W. Let $\mathcal{S}$ denote the set of all setpairs. Denote by $\delta(W) = \delta(W_t, W_h)$ the set of edges with one end-vertex in W_t and the other in W_h; note that a *cut* $\delta(W_t, V \setminus W_t)$ is the special case of $\delta(W)$ where the head and tail of W form a complementary pair of vertex sets.

The setpair formulation may be written as the following integer program (IP).

$$
\text{(IP)} \qquad \begin{array}{llcll}
\text{minimize} & \displaystyle\sum_{e \in E} c_e\, x_e & & & \\
\text{subject to} & \displaystyle\sum_{e \in \delta(W)} x_e & \geq & f(W), & \forall\, W \in \mathcal{S} \\
 & x_e & \in & \{0, 1\}, & \forall\, e \in E
\end{array}
$$

Here, f is some non-negative integer function on the setpairs, and $c : E \to \mathbb{R}_+$ gives the edge costs. Let (LP) denote the linear programming relaxation obtained from (IP) by replacing the constraints $x_e \in \{0, 1\}$ by $0 \leq x_e \leq 1$ for all edges $e \in E$..

Let H^* denote a k-VCSS of minimum cost, and let $z^* = c(H^*) = \sum_{e \in E(H^*)} c_e$ denote its cost. We now present the requirement function f used for the vertex connectivity problem. If W_t contains at least one vertex, say u, and W_h contains at least one vertex, say v, then note that H^* has at least $k - (n - |W_t \cup W_h|)$ edges in $\delta(W)$. This follows since on removing the vertices in $V - (W_t \cup W_h)$ from H^*, the resulting graph has at least this number of openly disjoint paths between u and v and each of these paths contributes one (or more) distinct edges to $\delta(W)$. (Two paths are called *openly disjoint* if every vertex that belongs to both paths is an end vertex of both paths.) Let $\mathcal{S}$ denote the set of all setpairs (W_t, W_h) such that $W_t \neq \emptyset$ and $W_h \neq \emptyset$. So let f be the following non-negative integer function defined on the setpairs.

$$
f(W) = \begin{cases} \max\{0,\ k - |V \setminus (W_h \cup W_t)|\}, & \text{if } W_t \neq \emptyset \text{ and } W_h \neq \emptyset \\ 0, & \text{otherwise} \end{cases}
$$

we denote by (LP-VC) the resultant linear programming relaxation. It will be convenient to keep a parameter ℓ, where ℓ is a positive integer, and denote by $P(\ell)$ the LP relaxation for the mincost ℓ-VCSS problem. Thus $P(k)$ (i.e. (LP-VC)) gives a lower bound $z(k)$, say, on the minimum cost of a k-VCSS.

Lemma 2.1. *Let $z^*(\ell)$ be the minimum cost of an ℓ-VCSS. Then $z^*(\ell) \geq z(\ell)$.* □

3. An O(Log k) Approximation for Undirected Graphs

3.1. k-outconnected graphs. A graph is said to be *k-outconnected* from a so-called *root* vertex r if there exist k openly disjoint paths from r to v, for each vertex v, $v \neq r$. The *mincost k-OC* problem is as follows: given an undirected graph $G = (V, E)$, a root vertex $r \in V$, and non-negative costs on the edges, find a minimum-cost subgraph H of G such that H is k-outconnected from r. This problem is NP-hard for $k \geq 2$.

Theorem 3.1 (Frank and Tardos (1989), Khuller and Raghavachari (1996)). *Let $G = (V, E)$, r, and $c : E \to \mathbb{R}_+$ be as above. There is a 2-approximation algorithm for the mincost k-OC problem. Moreover, the subgraph found by this algorithm has cost at most $2z(k)$.*

Proof. In the directed version $\widehat{G}$ of G, each edge e of G is replaced by two oppositely oriented arcs, and each of these two arcs has cost c_e. The LP relaxation of the directed mincost k-VCSS problem on $\widehat{G}$ has optimal value at most $2z(k)$. An optimal solution to this LP is also a feasible solution to the following LP relaxation $\widehat{P}$ of the directed mincost k-OC problem on $\widehat{G}$ (with any vertex as the root): There is a variable x_a for each arc a in $\widehat{G}$; let $\mathcal{R}$ denote the set of all setpairs $W = (W_t, W_h)$ such that the root r is in W_t and $W_h \neq \emptyset$; and for $W \in \mathcal{R}$ let $\hat{\delta}(W)$ denote the set of arcs (u, v) in $\widehat{G}$ with $u \in W_t$, $v \in W_h$.

$$
\begin{aligned}
\widehat{P}: \quad & \text{minimize} \sum_{a \in E(\widehat{G})} c_a\, x_a \\
& \text{subject to} \sum_{a \in \hat{\delta}(W)} x_a \;\geq\; k - (n - |W_t \cup W_h|), \qquad \forall\, W \in \mathcal{R} \\
& \qquad\qquad 0 \leq x_a \leq 1, \qquad \forall\, a \in E(\widehat{G})
\end{aligned}
$$

This LP $\widehat{P}$ has an integer optimal solution (see [4, Theorems 2.1, 2.2]). The Frank-Tardos algorithm finds a minimum-cost subdigraph $\widehat{H}$ that is k-outconnected from r, and the cost $c(\widehat{H})$ equals the optimal value of $\widehat{P}$, which is at most $2z(k)$. (The arc incidence vector of $\widehat{H}$ forms an optimal solution of $\widehat{P}$.) The undirected version of $\widehat{H}$ is a subgraph of G that is k-outconnected from r, and it has cost at most $2z(k)$. □

Remark. Our algorithm will apply this result to find a solution to the mincost ℓ-OC problem that has cost at most $2z(\ell)$, where $1 \leq \ell \leq k$.

3.2. 3-CRITICAL GRAPHS. For a graph G, let $\kappa(G)$ denote the *vertex connectivity*, i.e., the minimum number of vertices whose removal results in a disconnected graph or the trivial graph (namely, K_1). An *ℓ-separator* of a connected graph is a set of ℓ vertices whose removal results in a disconnected graph. A graph $G = (V, E)$ is called *3-critical* if the vertex connectivity decreases by $|S|$ on removing the vertices in any set S of at most three vertices, that is, if $\kappa(G - S) = \kappa(G) - |S|, \quad \forall S \subseteq V, |S| \leq 3$. If G is *not* 3-critical, then note that there exists a set S of three vertices such that no $\kappa(G)$-separator contains all the vertices in S.

Theorem 3.2 (Mader (1977)). *A 3-critical graph with vertex connectivity k has less than $6k^2$ vertices.*

Remark. Mader's proof [18] is written in German. He discusses this result in two survey papers written in English [19, 20].

3.3. THE ALGORITHM AND ITS ANALYSIS. The algorithm starts with $i := 1$, and a minimum-cost spanning tree H_1. Each iteration $i = 1, 2, \ldots,$ augments H_i to H_{i+1} by adding edges from $E(G) - E(H_i)$ such that the vertex connectivity increases by at least one, and the "augmenting cost" $c(H_{i+1}) - c(H_i)$ is approximately minimum. A detailed description of an iteration follows. Let $\ell = \kappa(H_i)$. If $\ell = k$, then we stop and output H_i as the desired k-VCSS. Now, suppose $\ell < k$. For each edge in H_i, we change the cost to zero (the other edges keep the original costs). By Mader's theorem (and the fact that n is at least $6k^2$) there exist three vertices such that no ℓ-separator of H_i contains all three vertices. We find three such vertices r_1, r_2, r_3 by exhaustively checking for each vertex set S of cardinality three whether $\kappa(H_i - S) > \ell - 3$. For each of these three vertices, we apply the Frank-Tardos algorithm with root r_j ($j = 1$, 2, or 3) and the modified edge costs to find a supergraph $H_{i,j}$ of H_i that is $(\ell+1)$-outconnected from r_j. We take (the edge set of) H_{i+1} to be the union of (the edge sets of) $H_{i,1}$, $H_{i,2}$, and $H_{i,3}$.

Lemma 3.3. *At every iteration $i = 1, 2, \ldots,$ we have $\kappa(H_{i+1}) \;\geq\; \kappa(H_i) \;+\; 1$.*

Proof. Let $\ell = \kappa(H_i)$. Note that $\ell < k$. Suppose that $\kappa(H_{i+1}) = \ell$. Then H_{i+1} has an ℓ-separator C, $C \subset V$. Now, H_i is not 3-critical by Mader's theorem, since $n \geq 6k^2 > 6\ell^2$. Hence, there exist three vertices in H_i such that for each ℓ-separator of H_i, at least one of these three vertices is absent from the ℓ-separator. The algorithm finds three such vertices r_1, r_2, r_3. W.l.o.g. r_1 is absent from C. The graph $H_{i,1}$, which is a subgraph of H_{i+1}, is $(\ell+1)$-outconnected from r_1. Hence, H_{i+1} has $(\ell+1)$ openly disjoint paths between r_1 and v, for every other vertex v, and one of these paths survives in $H_{i+1} - C$. We have a contradiction, since $H_{i+1} - C$ is connected. The lemma follows. □

Lemma 3.4. *At every iteration $i = 1, 2, \ldots,$ we have $c(H_{i+1}) - c(H_i) \leq \dfrac{6z(k)}{k-\ell}$, where $\ell = \kappa(H_i)$.*

Proof. Note that $\ell < k$. We will prove that for each of the three supergraphs $H_{i,j}$ ($j = 1$, 2, or 3) of H_i, the augmenting cost $c(H_{i,j}) - c(H_i)$ is at most $2z(k)/(k-\ell)$. Then the lemma follows immediately. Recall that (during the construction of $H_{i,j}$, $j = 1, 2, 3$) the edge costs are modified such that an edge already in H_i has zero cost, while the other edges have the original costs. Let $\boldsymbol{x} : E \to \mathbb{R}_+$ be an optimal solution to the linear program $P(k)$; note that the cost of $\boldsymbol{x}$ is $z(k)$. Let $\boldsymbol{x}' : E \to \mathbb{R}_+$ be given by

$$x'_e \;=\; \begin{cases} 1, & \text{if } e \text{ is in } H_i \\ \frac{x_e}{k-\ell}, & \text{otherwise.} \end{cases}$$

Clearly, $\boldsymbol{x}'$ has modified cost at most $z(k)/(k-\ell)$. We claim that $\boldsymbol{x}'$ is a feasible solution to the LP $P(\ell+1)$. Then, by Theorem 3.1, the Frank-Tardos algorithm finds an $(\ell+1)$-outconnected supergraph of H_i with augmenting cost at most $2z(k)/(k-\ell)$. To see the claim, consider any setpair $W \in \mathcal{S}$ and its constraint in the LP $P(\ell+1)$,

$$\sum_{e \in \delta(W)} x'_e \geq (\ell+1) - q,$$

where $q = n - |W_t \cup W_h|$. Suppose that H_i has no edges in $\delta(W)$; the other case can be handled similarly, but we skip the details. Then since H_i is ℓ-vertex connected, we have $q \geq \ell$. If $q \geq \ell+1$, then, obviously, $\boldsymbol{x}'$ satisfies this constraint. Otherwise, if $q = \ell$, then $\boldsymbol{x}'$ satisfies this constraint because (i) $\boldsymbol{x}$ satisfies the constraint of W in the LP $P(k)$, namely, $\sum_{e \in \delta(W)} x_e \geq k - \ell$, and (ii) each edge $e \in \delta(W)$ has $x'_e = x_e/(k-\ell)$. Our claim follows. □

Theorem 3.5. *Suppose that the input graph $G = (V, E)$ is k-vertex connected. Then the algorithm terminates with a k-VCSS that has cost at most $6(1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k})z(k)$, where $z(k)$ is the optimal value of the LP relaxation. The algorithm runs in time $O(k^2 n^4 (n + k^{2.5}))$.*

Proof. The vertex connectivity of H_i increases by at least one in every iteration, starting from one, so the algorithm terminates with a k-VCSS in at most $k-1$ iterations. The cost of the k-VCSS is at most

$$c(H_1) + \sum_{i=1}^{k-1} (c(H_{i+1}) - c(H_i)) \;\leq\; \frac{2z(k)}{k} + \sum_{\ell=1}^{k-1} \frac{6z(k)}{k-\ell} \;\leq\; 6(1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k})z(k).$$

To see the running time, note that each iteration i ($1 \le i < k$) runs the Frank-Tardos algorithm at most three times, and tests $\kappa(H_i - S)$ for at most n^3 sets of vertices S of cardinality three. Gabow's algorithm [9] tests the vertex connectivity κ in time $O((n+\kappa^{2.5}) \cdot \kappa n)$, and there is a version of the Frank-Tardos algorithm, due to Gabow, that runs in time $O(k^2 n^2 |E|)$, [8, Theorem 4.5]. □

4. Structure of a Basic Solution

To analyze the structure of a basic solution of the relaxation, we will need several definitions.

4.1. crossing setpairs. Given two setpairs $W = (W_t, W_h)$ and $Y = (Y_t, Y_h)$, let $W \otimes Y$ denote the setpair $(W_t \cup Y_t,\ W_h \cap Y_h)$, and let $W \oplus Y$ denote the setpair $(W_t \cap Y_t,\ W_h \cup Y_h)$. Two setpairs W and Y are said to be *comparable* if either (i) $W_t \supseteq Y_t$ and $W_h \subseteq Y_h$ or (ii) $W_t \subseteq Y_t$ and $W_h \supseteq Y_h$. The former case is denoted by $W \preceq Y$ and the latter case by $W \succeq Y$. Note that $W \preceq W \oplus Y$ and $W \succeq W \otimes Y$.

Two setpairs W and Y are said to be *non-crossing* if

(i) they are comparable, or (ii) their tails are disjoint, or (iii) their heads are disjoint.

Otherwise, W and Y are said to *cross*. Note that W and Y cross if and only if their tails intersect, their heads intersect and they are not comparable.

For a setpair $W = (W_t, W_h)$, let $\overline{W} = (W_h, W_t)$ be the *reverse setpair*. Observe that $\delta(W) = \delta(\overline{W})$ for an undirected graph. Also, note that $\overline{W \otimes Y} = \overline{W} \oplus \overline{Y}$. Two setpairs W and Y are said to be *overlapping* if

(i) W and Y cross, or (ii) $\overline{W}$ and Y cross (which is the same as W and $\overline{Y}$ cross).

Otherwise, the setpairs are called *non-overlapping*. A family of setpairs $\mathcal{L} \subseteq \mathcal{S}$ is called non-crossing (non-overlapping) if no two setpairs in $\mathcal{L}$ cross (overlap).

4.2. bisubmodular functions. A real-valued function f on $\mathcal{S}$ is called *bisubmodular* if for any two setpairs W and Y we have $f(W) + f(Y) \ge f(W \otimes Y) + f(W \oplus Y)$. A non-negative, integer-valued function f on $\mathcal{S}$ is called *crossing bisupermodular* if for any two crossing setpairs W and Y with $f(W) > 0$ and $f(Y) > 0$, we have $f(W) + f(Y) \le f(W \otimes Y) + f(W \oplus Y)$.

Let χ_W denote the edge incidence vector of $\delta(W)$. For any two setpairs W and Y, note that if an edge is present in $\delta(W \oplus Y)$ or $\delta(W \otimes Y)$, then it is present in $\delta(W)$ or $\delta(Y)$, and if an edge is present in both $\delta(W \oplus Y)$ and $\delta(W \otimes Y)$, then it is present in both $\delta(W)$ and $\delta(Y)$. Hence, we have $\chi_{W \otimes Y} + \chi_{W \oplus Y} \le \chi_W + \chi_Y$. Consequently, for any non-negative vector $\boldsymbol{x} : E \to \mathbb{R}_+$ on the edges, the corresponding function on setpairs, $x(\delta(W)) = \sum_{e \in \delta(W)} x_e$, is bisubmodular. (For any vector $\boldsymbol{x}$ on a groundset U and a subset Q of U, $x(Q)$ denotes $\sum_{i \in Q} x_i$.)

4.3. skew bisupermodular functions. A non-negative, integer-valued function f on $\mathcal{S}$ is called *skew bisupermodular* if for any two overlapping setpairs W and Y with $f(W) > 0$ and $f(Y) > 0$, we have

(i) $f(W) + f(Y) \le f(W \otimes Y) + f(W \oplus Y)$, or (ii) $f(W) + f(Y) \le f(\overline{W} \otimes Y) + f(\overline{W} \oplus Y)$.

This is motivated by problems on undirected graphs. For these problems, it turns out that the relevant functions f are symmetric, i.e., $f(W) = f(\overline{W})$, $\forall W \in \mathcal{S}$.

Note that every crossing bisupermodular function is skew bisupermodular. Also, every weakly supermodular symmetric function is skew bisupermodular. Thus the skew bisupermodular property is a generalization of these earlier notions.

4.4. CHARACTERIZING A BASIC SOLUTION VIA A NON-OVERLAPPING FAMILY. The main result of this section, Theorem 4.3, characterizes a basic solution of (LP) via a non-overlapping family of setpairs, assuming that the requirement function f is skew bisupermodular, and the given graph G is undirected. This result is an extension of classic results to our setting. Our proof is similar to earlier proofs in [13, Lemma 4.2] and [3, Theorem 2.1]; however, when we add a new setpair W to our non-overlapping family, we have to ensure that neither W nor $\overline{W}$ crosses a setpair in the family.

Given a feasible solution $\boldsymbol{x}$ to (LP), a setpair W is called *tight* if $x(\delta(W)) = f(W)$.

Lemma 4.1. *Let the requirement function f of (LP) be skew bisupermodular, and let $\boldsymbol{x}$ be a feasible solution to (LP) such that $x_e > 0$ for all edges $e \in E$. Suppose that the setpairs W and Y have $f(W) > 0, f(Y) > 0$, and moreover, W and Y overlap, and are tight (also, note that $\overline{W}$ is tight, it overlaps Y, and $f(\overline{W}) > 0$). Then one of the following holds:*

(i) The setpairs $W \otimes Y$ and $W \oplus Y$ are tight, and $\chi_W + \chi_Y = \chi_{W\otimes Y} + \chi_{W\oplus Y}$.

(ii) The setpairs $\overline{W} \otimes Y$ and $\overline{W} \oplus Y$ are tight, and $\chi_W + \chi_Y = \chi_{\overline{W}\otimes Y} + \chi_{\overline{W}\oplus Y}$.

Proof. Assume that $f(W) + f(Y) \le f(W \otimes Y) + f(W \oplus Y)$. The other case is similar. Then

$$\begin{aligned} f(W \otimes Y) + f(W \oplus Y) &\le x(\delta(W \otimes Y)) + x(\delta(W \oplus Y)) \\ &\le x(\delta(W)) + x(\delta(Y)) \\ &= f(W) + f(Y) \\ &\le f(W \otimes Y) + f(W \oplus Y) \end{aligned}$$

Hence, all the inequalities hold as equations, therefore, $W \otimes Y$ and $W \oplus Y$ are both tight. This proves the first part of (i). The second part of (i) follows from the following three facts: $\chi_W + \chi_Y \ge \chi_{W\otimes Y} + \chi_{W\oplus Y}$; $x_e > 0$ for all edges e; $x(\delta(W \otimes Y)) + x(\delta(W \oplus Y)) = x(\delta(W)) + x(\delta(Y))$. □

The next result is in [3, Lemma 2.3] and is used to prove the main result of this section.

Lemma 4.2. *Let W, Y and Z be setpairs. If Z crosses $W \otimes Y$ (or $W \oplus Y$) then either Z crosses W or Z crosses Y.* □

Theorem 4.3. *Let the requirement function f of (LP) be skew bisupermodular, and let $\boldsymbol{x}$ be a basic solution to (LP) such that $0 < x_e < 1$ for all edges $e \in E$. Then there exists a non-overlapping family $\mathcal{L}$ of tight setpairs such that:*
(i) Every setpair $W \in \mathcal{L}$ has $f(W) \ge 1$.
(ii) $|\mathcal{L}| = |E|$.
(iii) The vectors χ_W, $W \in \mathcal{L}$, are linearly independent.
(iv) $\boldsymbol{x}$ is the unique solution to $\{x(\delta(W)) = f(W), \forall W \in \mathcal{L}\}$.

Proof. Since $\boldsymbol{x}$ is a basic solution of (LP), and none of the constraints $0 \le x_e \le 1$ $(e \in E)$ holds with equality, there exist $|E|$ tight setpair constraints $x(\delta(W)) = f(W) > 0$ such that the edge incidence vectors χ_W corresponding to these constraints are linearly independent. Let $\mathcal{L}$ be a maximal, non-overlapping family of tight setpairs and let $span(\mathcal{L})$ denote the vector space spanned by the vectors χ_W, where $W \in \mathcal{L}$. The theorem holds because $span(\mathcal{L})$ equals $span(\mathcal{F})$, where $\mathcal{F}$ is the family of all tight setpairs. We prove this by contradiction. Suppose that there is a tight setpair Y such that $\chi_Y \notin span(\mathcal{L})$. Take such a Y that overlaps the minimum number of setpairs in $\mathcal{L}$ (this is a key step). Choose any setpair $W \in \mathcal{L}$ that overlaps Y; there exists such a W by the maximality of $\mathcal{L}$.

Either part (i) or part (ii) in the statement of Lemma 4.1 holds for W and Y. Assume that part (ii) holds. The other case is similar. Thus, the setpairs $\overline{W} \otimes Y$ and $\overline{W} \oplus Y$ are tight. In addition, $\chi_W + \chi_Y = \chi_{\overline{W} \otimes Y} + \chi_{\overline{W} \oplus Y}$. It follows that either $\chi_{\overline{W} \otimes Y} \notin span(\mathcal{L})$ or $\chi_{\overline{W} \oplus Y} \notin span(\mathcal{L})$. Consider the former case. Again, the other case is similar. Let $Q = \overline{W} \otimes Y$ and take any setpair $Z \in \mathcal{L}$. Suppose that Z overlaps Q. Then either Z crosses Q or Z crosses $\overline{Q} = \overline{\overline{W} \otimes Y} = W \oplus \overline{Y}$. Suppose that Z crosses Q. We apply Lemma 4.2 to Z and Q, noting that Z does not cross $\overline{W}$, to see that Z crosses Y. Similarly, if Z crosses $\overline{Q}$, then by Lemma 4.2, Z crosses $\overline{Y}$. Hence, Z overlaps Y. So every setpair in $\mathcal{L}$ that overlaps Q also overlaps Y.

It follows that Q overlaps fewer setpairs in $\mathcal{L}$ than Y. To see this, note that (1) W overlaps Y, by our choice of W, and (2) W does not overlap Q, because $Q \preceq \overline{W}$ (so W does not cross $\overline{Q}$) and W and Q are head disjoint (so W does not cross Q). Thus we get a contradiction to the choice of Y, since Q is a tight setpair that overlaps fewer setpairs in $\mathcal{L}$ than Y, and $\chi_Q \notin span(\mathcal{L})$. □

Remarks. The results in this section apply to undirected graphs but not to directed graphs, since we use the fact that a setpair W with $f(W) > 0$ is tight iff the reverse setpair $\overline{W}$ has $f(\overline{W}) > 0$ and is tight. In the theorem, if we replace a setpair $W \in \mathcal{L}$ by its reverse $\overline{W}$, then this preserves the properties (i)–(iv).

5. An Approximation Algorithm based on Iterative Rounding

The main result of this section is as follows.

Theorem 5.1. *Let k and n be positive integers, and let $\epsilon < 1$ be a positive number such that k is at most $(1-\epsilon)n$. There is a polynomial-time algorithm that, given an n-vertex (directed or undirected) graph, finds a solution to the k-vertex connectivity problem of cost at most $O(\sqrt{n/\epsilon})$ times the optimal cost.* □

The k-vertex connectivity requirement function f is crossing bisupermodular [5], hence, f is also skew bisupermodular. This can be seen as follows. For each setpair W, let $p(W)$ denote $|V \setminus (W_t \cup W_h)|$. For any two setpairs $W = (W_t, W_h)$ and $Y = (Y_t, Y_h)$, we have $p(W) + p(Y) = p(W \otimes Y) + p(W \oplus Y)$, because

$$|(W_t \cup Y_t) \cup (W_h \cap Y_h)| + |(W_t \cap Y_t) \cup (W_h \cup Y_h)| = |W_t| + |W_h| + |Y_t| + |Y_h|.$$

Hence, the above function $f(\cdot)$ is crossing bisupermodular (if either $W \otimes Y$ or $W \oplus Y$ has head or tail empty, then note that W, Y do not cross). The results in [5] imply that a basic optimal solution to (LP-VC) (if it exists) can be found in polynomial time, see [5], [12, Theorem 6.4.9]. Moreover, the requirement function f and (LP-VC) have the following "self-reducibility" property: for any edge e the "reduced" requirement function f' is also crossing bisupermodular (and skew bisupermodular), where f' is given by $f'(W) = f(W) - 1$ if the setpair W has $e \in \delta(W)$ and $f'(W) = f(W)$ otherwise (since the zero-one incidence function of e on $\mathcal{S}$ is bisubmodular).

Theorem 5.1 follows directly from the next result.

Theorem 5.2. *Let $\epsilon < 1$ be a positive number, and suppose that $k \leq (1-\epsilon)n$. Then any nonzero basic solution of (LP-VC) has an edge of weight $\Omega(\sqrt{\frac{\epsilon}{n}})$.*

We give the proof for undirected graphs, using results on skew bisupermodular functions. After that, we sketch the proof for directed graphs via crossing bisupermodular functions. The next result is from [3, Theorem 1.1].

Theorem 5.3. *Suppose that the requirement function f for the linear program (LP-VC) is crossing (or, skew) bisupermodular. Let $\boldsymbol{x}$ be a nonzero basic solution of (LP-VC), and let $\mathcal{L}$ be a non-crossing family of setpairs characterizing $\boldsymbol{x}$. Then there exists an edge e with $x_e \geq \frac{1}{\Omega(\sqrt{|\mathcal{L}|})}$.* □

Focus on a nonzero basic solution $\boldsymbol{x}$ of (LP-VC). By Theorem 4.3, $\boldsymbol{x}$ corresponds to a non-crossing family of setpairs $\mathcal{L}$ such that each setpair has a nonempty head and a nonempty tail. We may assume that each setpair $W \in \mathcal{L}$ has $|W_h| \geq |W_t|$, else we may replace W by $\overline{W}$ and this preserves all the properties of $\mathcal{L}$ in Theorem 4.3. Let $\mathcal{P}$ denote the poset (and the Hasse diagram) representing $\mathcal{L}$, and recall that $\mathcal{P}$ has a node W for each setpair W in $\mathcal{L}$ and it has an arc (W, Y), where $Y \in \mathcal{L}$, if $W \preceq Y$ and there is no other $Z \in \mathcal{L}$ such that $W \preceq Z \preceq Y$ (we omit arcs implied by transitivity from $\mathcal{P}$); we call W a *child* of Y, and call Y a *parent* of W. (Note that the terms "node" and "arc" refer to the poset $\mathcal{P}$, and the terms "vertex" and "edge" refer to the input graph or digraph G.) A *diamond* of a poset is a set of four nodes W, X, Y, Z such that X and Y are incomparable, W is a common descendant of X, Y, and Z is a common ancestor of X, Y. A poset is called *diamond-free* if it contains no diamond. The poset $\mathcal{P}$ representing $\mathcal{L}$ is diamond-free, see [3, Lemma 3.1].

Lemma 5.4. *Suppose that $k \leq (1-\epsilon)n$. Then there are at most $\frac{2}{\epsilon}$ pairwise incomparable setpairs in $\mathcal{L}$ such that all their tails contain a common vertex.*

Proof. Let the setpairs $Y^1, Y^2, \ldots, Y^q \in \mathcal{L}$ be pairwise incomparable, and let v be a vertex that occurs in the tail of each of these setpairs. Since the Y^i are incomparable and their tails intersect, their heads are disjoint. Each head Y_h^i has cardinality at least $\frac{1}{2}(n-k) \geq \frac{1}{2}\epsilon n$; this follows from the properties of $\mathcal{L}$, and the fact that each requirement $f(Y^i)$ is at least one. Hence, there are at most $\frac{2}{\epsilon}$ such setpairs. □

We partition the setpairs in $\mathcal{L}$ into several sets.

- W is a type I setpair if it has no parents in $\mathcal{P}$.
- W is a type II setpair if it has at least two parents in $\mathcal{P}$.
- W is a type III setpair if it has exactly one parent Y in $\mathcal{P}$, and $W_t = Y_t$.
- W is a type IV setpair if it has exactly one parent Y in $\mathcal{P}$, and $W_t \neq Y_t$.

Lemma 5.5. *Suppose that $k \leq (1-\epsilon)n$. (a) Then there are at most $\frac{2n}{\epsilon}$ setpairs of type IV. (b) Moreover, each setpair (node) in $\mathcal{P}$ has at most $\frac{2n}{\epsilon}$ children.*

Proof. We prove (a) first, then sketch the proof of (b). To each type IV setpair W with parent Y we assign a vertex v in $W_t \setminus Y_t$. We claim that a vertex can be assigned to at most $\frac{2}{\epsilon}$ setpairs of type IV. This will imply the claim. Take a vertex v that is assigned to the type IV setpairs $W^1, W^2, \ldots, W^q$. Let their parents be $Y^1, Y^2, \ldots, Y^q$, respectively. Now $v \in W_t^1 \cap W_t^2 \cap \cdots \cap W_t^q$, but $v \notin Y_t^1 \cup Y_t^2 \cup \cdots \cup Y_t^q$. The setpairs $W^1, W^2, \ldots, W^q$ associated with v must be pairwise incomparable (since v is in the tail of every W^i, but for every proper ancestor of a W^i note that v is not in the tail). Then by Lemma 5.4 applied to $W^1, W^2, \ldots, W^q$, there are at most $\frac{2}{\epsilon}$ such setpairs whose tails contain v.

For (b), consider all the children $Y^1, Y^2, \ldots, Y^q$ of a node $W \in \mathcal{P}$. Then $Y^1, Y^2, \ldots, Y^q$ are pairwise incomparable (otherwise, one of the arcs (Y^i, W) in $\mathcal{P}$ is redundant), and moreover, each of the tails Y_t^i, $i = 1, \ldots, q$, contains W_t which is nonempty. Then by Lemma 5.4 applied to $Y^1, Y^2, \ldots, Y^q$, we have $q \leq \frac{2}{\epsilon}$. □

Proof of Theorem 5.2. Let m denote $|\mathcal{L}|$. If $m < 4n/\epsilon$, then by Theorem 5.3 there is an edge e with $x_e \geq \sqrt{\frac{\epsilon}{\Theta(n)}}$. Hence, in this case, Theorem 5.2 holds. The rest of the proof focuses on the other case, $m \geq 4n/\epsilon$, and shows that there exists an edge e with either $x_e \geq \frac{1}{\Theta(1)}$ or $x_e \geq \Omega(\epsilon) \geq \Omega(\sqrt{\frac{\epsilon}{n}})$. (Note that $k \leq n-1$, hence $\epsilon \geq \frac{1}{n}$, and so $\epsilon \geq \sqrt{\frac{\epsilon}{n}}$.) Thus we have $|\mathcal{L}| = m \geq \frac{4n}{\epsilon}$. Hence, by Lemma 5.5, $\mathcal{L}$ has at least $\frac{m}{6}$ setpairs of one of the types I, II, or III. We consider each possibility in turn.

(Type I) Suppose that there are at least $\frac{m}{6}$ type I setpairs. The type I setpairs are pairwise incomparable. Hence the edge sets $\delta(W)$, where W is a type I setpair, are *quasidisjoint*, i.e., for each edge uv, there are at most two setpairs W, Y (among the type I setpairs) such that $uv \in \delta(W)$ (say $u \in W_t$ and $v \in W_h$) and $uv \in \delta(Y)$ (say $v \in Y_t$ and $u \in Y_h$). Hence, there is a type I setpair $W \in \mathcal{L}$ such that $\delta(W)$ has size at most 12, and so there is an edge $e \in \delta(W)$ with $x_e \geq \frac{1}{12}$.

(Type II) Suppose that there are at least $\frac{m}{6}$ type II setpairs. Let $Y^1, \ldots, Y^q$ be the minimal nodes in $\mathcal{P}$ (those with no children); clearly, these nodes are pairwise incomparable. Each Y^i and its ancestors forms a (directed) tree (since $\mathcal{P}$ is diamond-free). Each node of $\mathcal{P}$ is in at most $\frac{2}{\epsilon}$ of these trees, by Lemma 5.4. Focus on one of the trees and note that the number of leaf nodes (maximal nodes of $\mathcal{P}$) of the tree is greater than or equal to the number of type II nodes in the tree. Summing over all the trees, we see that the total number of leaf nodes (maximal nodes of $\mathcal{P}$) is at least $\frac{\epsilon}{2}$ times the number of type II nodes in $\mathcal{P}$, hence, $\mathcal{P}$ has at least $\frac{m\epsilon}{12}$ maximal nodes; clearly, these nodes are pairwise incomparable. Then the edge sets $\delta(W)$, where W is a maximal node of $\mathcal{P}$, are quasidisjoint, so there is an edge e with $x_e \geq \frac{\epsilon}{24}$.

(Type III) Finally, suppose that there are at least $\frac{m}{6}$ type III setpairs. Consider all of the type III setpairs $W^1, W^2, \ldots W^q$ and their unique parents $Y^1, Y^2, \ldots Y^q$. Observe that $f(W^i) + 1 \leq f(Y^i)$, since $|W_t \cup W_h| < |Y_t \cup Y_h|$. Moreover, if W^i and W^j (where $1 \leq i < j \leq q$) have distinct parents Y^i and Y^j, then the edge sets $\delta(Y^i) \setminus \delta(W^i)$ and $\delta(Y^j) \setminus \delta(W^j)$ are quasidisjoint. To see this, consider an edge $uv \in \delta(Y^i) \setminus \delta(W^i)$ with $u \in Y^i_t = W^i_t$ and $v \in Y^i_h - W^i_h$. Now, every proper ancestor Z of W^i has $v \in Z_h$, so if W^j is an ancestor of W^i, then we cannot have $uv \in \delta(Y^j) \setminus \delta(W^j)$, $u \in Y^j_t = W^j_t$, and $v \in Y^j_h - W^j_h$. Consequently, by Lemma 5.5(b), any edge occurs in at most $\frac{4}{\epsilon}$ of the edge sets $\delta(Y^i) \setminus \delta(W^i)$, and moreover, $x(\delta(Y^i) \setminus \delta(W^i)) \geq 1$, for each $i = 1, \ldots, q$. Hence, there is an edge e with $x_e \geq \frac{\epsilon}{24}$.

This concludes the proof of Theorem 5.2 for undirected graphs: any nonzero basic solution of (LP-VC) has an edge of weight $\Omega(\sqrt{\epsilon/n})$.

The proof extends to directed graphs via Theorem 5.3. We partition the non-crossing family of setpairs $\mathcal{L}$ into two subfamilies $\mathcal{L}_1$ and $\mathcal{L}_2$, depending on whether or not a setpair $W \in \mathcal{L}$ has $|W_h| \geq |W_t|$. Suppose that $|\mathcal{L}_1| \geq |\mathcal{L}_2|$; otherwise, we use a symmetric argument. Then we apply the arguments for undirected graphs to $\mathcal{L}_1$ (partitioning it into types I, II, III, IV, etc.). We obtain similar lower bounds on $\max\{x_e\}$, the main difference being that we lose a factor of two in the lower bound (since we have $|\mathcal{L}_1| \geq \frac{m}{2}$ rather than $|\mathcal{L}| = m$). Hence, Theorem 5.2 holds for directed graphs. □

Theorem 5.1 follows from Theorem 5.2: for $k \leq (1-\epsilon)n$, iterative rounding gives an $O(\sqrt{n/\epsilon})$ approximation algorithm for the (directed or undirected) k-vertex connectivity problem.

Remark. There is an example of the undirected problem with an extreme point x of (LP-VC) such that each edge e has $x_e \leq \frac{1}{\Omega(\sqrt{k})}$. Hence, in Theorem 5.2, the lower bound on the maximum weight of an edge cannot be improved beyond $\frac{1}{\Omega(\sqrt{k})}$. Note, however, that this does not imply a similar lower bound on the integrality gap of the linear program.

References

[1] V.Auletta, Y.Dinitz, Z.Nutov and D.Parente, "A 2-approximation algorithm for finding an optimum 3-vertex-connected spanning subgraph", *Journal of Algorithms*, **32**, 1999, pp. 21–30.

[2] J.Cheriyan, T.Jordan and Z.Nutov, "On rooted node-connectivity problems", *Algorithmica*, **30**, 2001, pp. 353–375.

[3] J. Cheriyan and S. Vempala, "Edge coverings of setpairs and the iterative rounding method", submitted for journal publication, September 2001. Preliminary version in the *Proceedings of the 8th International Integer Programming and Combinatorial Optimization Conference*, Utrecht, The Netherlands, pp. 30-44, 2001.

[4] A. Frank, "Connectivity augmentation problems in network design", in *Mathematical Programming: State of the Art 1994*, (Eds. J. R. Birge and K. G. Murty), pp. 34–63, The University of Michigan, Ann Arbor, MI, 1994.

[5] A. Frank and T. Jordan, "Minimal edge-coverings of pairs of sets", *Journal of Combinatorial Theory, Series B*, **65**, pp. 73-110, 1995.

[6] A.Frank and E.Tardos, "An application of submodular flows", *Linear Algebra and its Applications*, **114/115**, 1989, pp. 329-348.

[7] L. Fleischer, K. Jain, and D. P. Williamson, "An iterative rounding 2-approximation algorithm for the element connectivity problem," to appear in *Proc. 42nd IEEE FOCS*, October 2001.

[8] H.N.Gabow, "A representation for crossing set families with applications to submodular flow problems", *Proc. 4th Ann. ACM-SIAM Symposium on Discrete Algorithms*, 1993, pp. 202-211.

[9] H.N.Gabow, "Using expander graphs to find vertex connectivity", *Proc. 41st IEEE FOCS*, Nov. 2000.

[10] M.Goemans, A.Goldberg, S.Plotkin, D.Shmoys, E.Tardos and D.Williamson, "Improved approximation algorithms for network design problems", *Proc. 5th Ann. ACM-SIAM Symposium on Discrete Algorithms*, 1994, pp. 223-232.

[11] M. Goemans and D. Williamson, "A general approximation technique for constrained forest problems", *SIAM Journal on Computing*, **24**, pp. 296-317, 1995.

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

[13] K. Jain, "A factor 2 approximation algorithm for the generalized Steiner network problem", *Combinatorica*, **21**(1), pp. 39-60, 2001. Preliminary version in *Proc. 39th IEEE FOCS*, 1998.

[14] K.Jain, I.Mandoiu, V.V.Vazirani and D.P.Williamson, "A primal-dual schema based approximation algorithm for the element connectivity problem", *Proc. 10th Ann. ACM-SIAM Symposium on Discrete Algorithms*, 1999, pp. 484-489.

[15] S.Khuller and B.Raghavachari, "Improved approximation algorithms for uniform connectivity problems", *Journal of Algorithms*, **21**, 1996, pp. 434–450.

[16] S. Khuller and U. Vishkin, "Biconnectivity Approximations and Graph Carvings", *Journal of the ACM*, **41(2)**, pp. 214–235,1994. Preliminary version in *Proc. 24th STOC*, 1992.

[17] G.Kortsarz and Z.Nutov, "Approximating node connectivity problems via set covers", *Approximation Algorithms for Combinatorial Optimization*, Third international workshop, APPROX 2000, (K.Jansen, S.Khuller (eds.)), Springer, LNCS **1913**, 2000, pp. 194-205.

[18] W.Mader, "Endlichkeitsätze für k-kritische Graphen", (German), *Math. Ann.*, **229**, 1977, pp. 143–153.

[19] W.Mader, "Connectivity and edge-connectivity in finite graphs", *Surveys in Combinatorics (Proc. Seventh British Combinatorial Conf., Cambridge, 1979)*, pp. 66–95, London Math. Soc. Lecture Note Ser., 38, Cambridge Univ. Press, Cambridge-New York, 1979.

[20] W.Mader, "On k-critically n-connected graphs", *Progress in Graph Theory* (Waterloo, Ont., 1982), pp. 389–398, Academic Press, Toronto, ON, 1984.

[21] R.Ravi and D.P.Williamson, "An approximation algorithm for minimum-cost vertex-connectivity problems", *Algorithmica*, **18**, 1997, pp. 21-43. (Preliminary version in *Proc. 6th Ann. ACM-SIAM Symposium on Discrete Algorithms*, 1995, pp. 332-341.)

[22] E. Tardos, "A strongly polynomial algorithm to solve combinatorial linear programs", *Operations Research*, **34**, pp. 250-256, 1986.