The Vital Core Connectivity Problem

Sylvia Boyd and Amy Cameron*

School of Information Technology & Engineering; Dept. of Mathematics & Statistics

University of Ottawa, Ottawa, Ontario, K1N 6N5, Canada

Technical Report, University of Ottawa, 2011†

Abstract

Let $G = (V, E)$ be an edge-weighted complete graph representing a network in which the edges represent potential links, and the vertices (centres) are partitioned into two classes – vital vertices, which represent the vital core of the network, and secondary vertices. We consider the *vital core connectivity problem* (VCC), which is the problem of finding a minimum weight spanning multi-subgraph of G which is k-edge connected overall and whose vital core remains at least l-edge connected even if some or all of the secondary vertices are removed. The VCC arises naturally in many practical applications in which one wishes to design a network at minimum cost which will not only survive the loss of a certain number of links overall, but for which the vital core remains at least l-edge connected even if some or all of the secondary centres are lost. We show that the VCC is, in general, NP-hard, and present the first constant factor approximation algorithm for this problem, as well as give an upper bound on the integrality gap of its linear programming relaxation. In particular, we show an approximation guarantee (and upper bound on the integrality gap) of $\frac{8}{3}$ for $l \geq \lceil \frac{k}{2} \rceil$, $\frac{19}{6}$ for $2 \leq l < \lceil \frac{k}{2} \rceil$, and $\frac{5}{2}$ for $l = 1$. We also show that in the case that $k = l = 1$, the VCC results in a useful generalization of the minimum cost spanning tree problem, for which we provide a complete linear description as well as a polynomial-time algorithm. Lastly, we provide a complete linear description of, and a polynomial-time algorithm for, an extension of a special case of VCC to numerous disjoint vital cores.

*Sylvia Boyd: sylvia@site.uottawa.ca, Amy Cameron: acame097@uottawa.ca

†Published in: A. Cameron, Approximation Algorithms for Network Connectivity Problems, Ph.D. Thesis, University of Ottawa (2012).

1 Introduction

Given a network consisting of vital and secondary centres, specific non-negative costs for connecting any two centres with a link, and non-negative integers $k \geq 1$, $l \leq k$, we examine finding a cheapest way to construct a network so that it is k-edge connected (i.e., so that the network remains connected even if any $k-1$ of the links are lost), and so that if some or all of the secondary centres are lost, the vital core of the network (consisting of the vital centres) remains l-edge connected. We call this problem the *vital core connectivity problem*.

This problem has a number of very useful applications in the construction of survivable networks. For example, military outposts and/or remote transmitters are often placed in key locations such as mountain ranges, etc.. Since these outposts are often more vulnerable and susceptible to being lost through enemy attack or weather, it is useful to have a communications network that is capable of surviving the loss of a certain number of lines without being disconnected, as well as capable of surviving the loss of any or all of the vulnerable (secondary) outposts and still have the vital core remain connected. This holds as well for subway systems. Subway stations with a high number of subway lines joined to them are more vulnerable to targeted attack [1]. Thus, it is desirable for a subway network to be constructed in such a way that not only is it able to sustain the loss of a certain number of subway lines, but also that its vital core remains at least connected if some or all of the more vulnerable (secondary) stations are lost. Such a network makes serious disruption of the subway system through targeted attack much more difficult. Usefulness to have such a network can be seen with the March 2004 Madrid and July 2005 London subway bombings [1, 21]. Notice that finding a solution that is either optimal or very close to optimal means substantial financial savings for companies constructing such networks. Notice as well that in some applications, it is useful to allow more than one link (i.e., multi-links) to be built between a pair of centres in order to build a reliable network at lower cost. Instances of this occur in the laying of underwater cables between islands and a mainland, where a link failure is considered to be the failure of a cable.

A graph is *k-edge connected* (respectively, *k-vertex connected*) if it has at least k edge-disjoint (respectively, internally vertex-disjoint) paths between every pair of vertices; i.e. if any $k-1$ edges (respectively, vertices) are removed, the graph is still connected. Given a complete graph G with non-negative edge costs, the *k-edge connected spanning subgraph problem* (kEC) is the problem of finding a minimum cost k-edge connected spanning subgraph of G. When multiple copies of edges are allowed in the solution, we denote the problem by multi-kEC. The problem of finding a minimum cost k-vertex connected spanning subgraph of G is denoted kVC.

We can now more formally define our problem. The *vital core connectivity* *problem* (VCC) is defined as follows: Given a complete graph, $G = (V, E)$, with non-negative edge costs $c \in \mathbb{R}^E$, non-negative integers $k, l \in \mathbb{Z}$, $k \geq l \geq 1$, and the vertices partitioned into vital vertices V^* and secondary vertices $V \backslash V^*$, find a minimum cost k-edge connected spanning multi-subgraph of G such that the subgraph restricted to V^* is l-edge connected. We denote a particular instance of VCC by VCC(k, l, V^*). As we will see, the VCC is, in general, an NP-hard problem and is a generalization of the minimum spanning tree (MST) and k-edge connected (kEC) problems. Although this problem arises naturally in many applications, and has some similarities with other problems, to the best of our knowledge we present the first study of it.

Special cases of VCC that are worth noting are as follows: When $V^* = V$, the special case VCC$(1, 1, V)$ is precisely the minimum spanning tree problem. The more general special case VCC$(1, 1, V^*)$ will be referred to as the *extended minimum spanning tree problem* (EMST), and will be examined separately in this paper. It is the problem of finding a minimum cost spanning tree of G whose subgraph induced by V^* is also a spanning tree of the vital core $G[V^*]$. The special case VCC$(k, 1, V^*)$ (i.e., when $l = 1$) is the problem of finding a minimum cost k-edge connected spanning multi-subgraph of G such that the subgraph restricted to the set of vital vertices V^* is connected; in particular, such that the secondary vertices do not form a cut set (i.e., a subset of vertices whose removal from a connected graph disconnects that graph). When $l = 1$ and the set of secondary vertices $V \backslash V^*$ consists of a single vertex $r \in V$, then the problem is that of finding a minimum cost k-edge connected spanning multi-subgraph of G in which r is not a cut vertex (i.e., a vertex whose removal from a connected graph disconnects that graph). As we will see, the special case VCC(k, l, V), i.e. where $V^* = V$, is precisely the problem of finding a minimum cost k-edge connected spanning multi-subgraph of G; i.e., it is precisely multi-kEC.

As it is considered highly unlikely that efficient algorithms for exactly solving NP-hard problems exist, we look at designing efficient algorithms that return approximate, or near-optimal, solutions. If an algorithm runs in polynomial time and returns a solution whose value is always within a constant factor α of the optimal value, then the algorithm is known as an *α-approximation algorithm*, and the factor α is called the *approximation guarantee* of the algorithm. A related concept to approximation guarantees is that of integrality gaps. Given a minimization problem P, its integer linear program Q, and its linear programming (LP) relaxation Q_{LP}, the *integrality gap* for the linear programming relaxation is the largest ratio $opt(Q)/opt(Q_{LP})$ over all possible edge costs. If the integrality gap of a problem's LP relaxation is 1, then there always exists an optimal solution of its LP formulation that is integer. The integrality gap provides a measure of the quality of the lower bound provided by Q_{LP} for Q. Moreover, a polynomial-time constructive proof for a bound α on the integrality gap provides an α-approximation algorithm for problem P.

In this paper, we present the first constant factor approximation algorithm for VCC(k, l, V^*), as well as the first upper bound for the integrality gap for its linear programming relaxation. For VCC(k, l, V^*), we show an approximation guarantee (and upper bound on the integrality gap) of $\frac{8}{3}$ for $l \geq \lceil \frac{k}{2} \rceil$, $\frac{19}{6}$ for $2 \leq l < \lceil \frac{k}{2} \rceil$, and $\frac{5}{2}$ for $l = 1$ or for $l = \lceil \frac{k}{2} \rceil$ and k even. When $l = \lceil \frac{k}{2} \rceil$ and k is odd the approximation guarantee and upper bound for the integrality gap are at most $\frac{8}{3}$ and asymptotically approach $\frac{5}{2}$ from above as k gets large. For the special case VCC$(2, 1, V^*)$ (in which the secondary vertices are not a cut set), our algorithm has a $\frac{3}{2}$ approximation guarantee, generalizing our previous $\frac{3}{2}$-approximation algorithm for multi-2EC [3]. In particular, when there is only one secondary vertex $r \in V$, VCC$(2, 1, V \backslash \{r\})$ has a $\frac{3}{2}$ approximation guarantee and finds a feasible multi-2EC solution in which r is not a cut vertex. For the special case VCC(k, k, V^*), $V^* \neq V$, our approximation guarantee is 2; while for the special case VCC(k, l, V) (which is simply multi-kEC), we have an approximation algorithm with approximation guarantee of $\frac{3}{2}$ when k is even, and $\frac{3}{2} + \frac{1}{2k}$ when k is odd (agreeing with our previous results [3]). We also show that in the case that $k = l = 1$, VCC results in a useful generalization (EMST) of the minimum cost spanning tree problem; a generalization for which we provide a complete linear description as well as a polynomial-time algorithm. This polyhedral description and algorithm for EMST will be used as an essential tool for the approximation algorithm and integrality gap results for the general VCC. We also show that EMST can be extended naturally to a problem on numerous disjoint vital cores, and provide a complete linear description of, and a polynomial-time algorithm for, this problem.

The VCC is related to, but distinct from, other connectivity problems, some of which we outline here.

VCC and k-vertex Connectivity

While not guaranteeing vertex connectivity, there are instances of VCC that, in particular, guarantee that a given subset $V' \subset V$, $|V'| < k$, of vertices is not a cut set. This is obtained by the instance VCC$(k, 1, V \backslash V')$, $|V'| < k$; i.e., where the set of secondary vertices is V'. On the other hand, kVC guarantees that the removal of any $k - 1$ vertices does not disconnect the graph. In particular, VCC$(2, 1, V \backslash \{r\})$ finds a minimum cost 2-edge connected spanning multi-subgraph in which $r \in V$ is not a cut vertex; while 2VC finds a minimum cost simple spanning subgraph that has no cut vertices. Notice that kVC is a special case of VCC$(k, 1, V \backslash V')$, $|V'| < k$. The best known approximation algorithm for kVC has an approximation guarantee of 2 [16]. For VCC$(k, 1, V \backslash V')$, we provide an approximation guarantee of $\frac{5}{2} - \frac{2}{k}$ for k even, and $\frac{5}{2} - \frac{3}{2k}$ for k odd.

VCC and Multi-k-edge Connectivity

The VCC is a generalization of multi-kEC, since, as we will see, multi-kEC is precisely VCC(k, l, V), i.e., the case where the set of vital vertices is V. The current best general constant factor approximation guarantee for multi-kEC and

kEC is 2, from Kamal Jain [14]. In addition to providing an approximation guarantee, Jain's approximation algorithm provides a constructive proof that the integrality gap of the LP relaxation of kEC and multi-kEC is at most 2. There exist approximation algorithms that, as a special case, for multi-kEC have an approximation guarantee (and integrality gap) of $\frac{3}{2}$ when k is even, and $\frac{3}{2} + \frac{1}{2k}$ when k is odd [3, 13].[1] In this special case, our approximation guarantee and integrality gap matches these results.

VCC and Element Connectivity Problem

In addition to allowing edge failures, VCC allows certain vertex failures. Thus, the *element connectivity problem* (ELC-SNDP) [15], which is also an intermediate problem between vertex and edge connectivity, is a problem that is related to VCC. Let $G = (V, E)$ be a graph with non-negative edge costs $c \in \mathbb{R}^E$, $T \subset V$ be a subset of vertices called terminals, and r_{uv} be a non-negative integer connectivity requirement for each pair of distinct vertices $u, v \in T$. Elements are defined as the edges in E and the vertices in $V \backslash T$. ELC-SNDP is the problem of finding a minimum cost (simple) subgraph of G such that there exist at least r_{uv} element-disjoint paths between every distinct pair of vertices $u, v \in T$. The best known approximation guarantee for ELC-SNDP is 2 [9]. Although both VCC and ELC-SNDP allow vertex and edge failures, the problems are clearly different: Let $T = V^*$ and let $r_{ij} = l$ for all $i, j \in V^*$. A feasible solution of ELC-SNDP is such that between all distinct pairs of vital vertices $i, j \in V^*$, there are at least l edge-disjoint paths that are disjoint with respect to the secondary vertices $V \backslash V^*$. There is no guarantee regarding the overall edge connectivity of the subgraph. On the other hand, a feasible solution of VCC(k, l, V^*) is k-edge connected overall, but also is such that between all distinct pairs of vital vertices $i, j \in V^*$, there are at least l edge-disjoint paths that do not contain any vertices from $V \backslash V^*$.

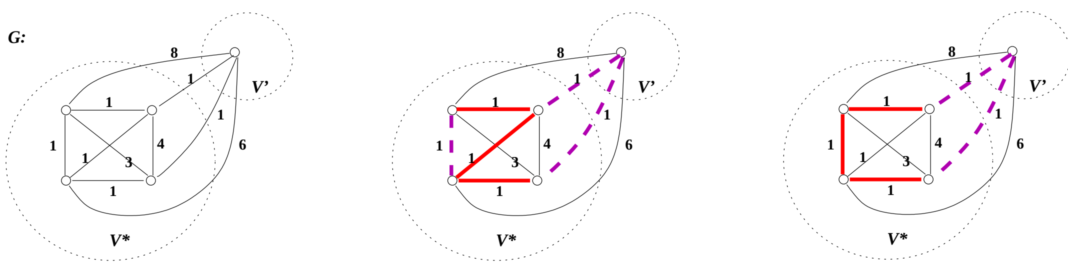
VCC and Augmentation Problems

Given a, possibly disconnected, multi-subgraph $H' = (V, E')$ of a weighted graph $G = (V, E)$, and a non-negative integer k, the *multi-edge connectivity augmentation problem* (multi-AUG) is the problem of finding a minimum cost set of edges $F \subseteq E$ such that $H' + F$ is a k-edge connected spanning multi-subgraph of G. The set F may contain edges from E', and may also contain multiple copies of edges. Although Jain does not explicitly mention it, a 2-approximation algorithm can be obtained for multi-AUG from Jain's 2-approximation algorithm for his problem [14].[2] There is a $\frac{3}{2}$-approximation algorithm for instances of multi-AUG in which H' is a simple, connected, unweighted graph and $k = 2$ [8]. In the special case where H' consists of the secondary vertices $V \backslash V^*$ plus an optimal multi-l-edge connected spanning subgraph of $G[V^*]$, the solution to multi-AUG

[1] There is also a $\frac{3}{2}$-approximation algorithm for 2EC and 2VC when the edge costs are metric (satisfy the triangle inequality) [10].

[2] This can be obtained by simply applying Jain's problem to the survivable network design problem and adjusting the cut constraints by the appropriate amount.

is a feasible solution to $\text{VCC}(k, l, V^*)$. However, as shown in Figure 1, such a solution, while being a feasible solution to $\text{VCC}(k, l, V^*)$, is not necessarily an optimal solution to $\text{VCC}(k, l, V^*)$. In Figure 1(a), we show a complete weighted graph on 5 vertices. In Figure 1(b) and 1(c), we find different optimal 1-edge connected subgraphs of $G[V^*]$ (the subgraphs formed by the red, bold edges) and respectively solve multi-AUG with $k = 2$ (the dashed edges are the edges added in the minimum cost augmentation). Clearly, the multi-AUG solutions shown in Figure 1(b) and (c) are both feasible solutions to $\text{VCC}(2, 1, V^*)$, but only Figure 1(c) is an optimal solution to $\text{VCC}(2, 1, V^*)$. Thus, contrary to what might be expected, finding a minimum cost multi-l-edge connected spanning subgraph on $G[V^*]$, and then augmenting it at minimum cost to a multi-k-edge connected spanning subgraph on G, is not the same problem as $\text{VCC}(k, l, V^*)$.



(a) (b) (c)

total cost = 3 + 3 = 6

opt(VCC(2,1,V) on G) = 3 + 2 = 5*

Figure 1: Example showing that finding a minimum cost multi-l-edge connected spanning subgraph on $G[V^*]$, and then augmenting it at minimum cost to a multi-k-edge connected spanning subgraph on G, does not always yield an optimal solution to $\text{VCC}(k, l, V^*)$.

An outline of this paper is as follows: In Section 2 we show that $\text{VCC}(2, 1, V^*)$, and thus $\text{VCC}(k, l, V^*)$, is NP-hard, but that there exists a polynomial-time algorithm that solves EMST. We also give a complete linear description for EMST. In Section 3.1, we present the results necessary for our approximation algorithm and integrality gap results for the general $\text{VCC}(k, l, V^*)$. In Section 3.2, we present Algorithm VCC, an approximation algorithm for $\text{VCC}(k, l, V^*)$; and present an upper bound on the problem's LP relaxation. In Section 3.3, we present the results for special cases of VCC, as well as present special cases in which the approximation guarantee and upper bound of the integrality gap can be further improved; in particular, when $l = 1$. Both the algorithm and polyhedral description of EMST in Sections 2.1 and 2.2 are fundamental to Section 3. We conclude the section by making some final observations and comparisons. In the final section, Section 4, we provide a complete linear description of, and a polynomial-time algorithm for, an extension of EMST to numerous disjoint vital cores.

We conclude this introduction with some notation. Given a minimization problem $\mathcal{P}$, its integer linear program $\mathcal{Q}$, and its linear programming relaxation $\mathcal{Q}_{LP}$, let $opt_{\mathcal{G}}(\mathcal{Q})$ be the total edge cost of an optimal solution to the linear programming problem $\mathcal{Q}$ on the graph $G = (V, E)$, and let $opt_G(\mathcal{Q}_{LP})$ be the total edge cost of an optimal solution to the linear programming relaxation $\mathcal{Q}_{LP}$ on G. Let $cost_G(\mathcal{H})$ be the total edge cost of a feasible solution $\mathcal{H}$ to problem $\mathcal{Q}$. Note that the subscript G will be omitted when it is clear what graph we are discussing. For $V' \subset V$, let $\gamma_G(V')$ be the edge set consisting of the edges of G with both ends in V', and $\delta_G(V')$ be the edge set consisting of the edges of G with exactly one end in V'. Let $G[V']$ be the subgraph of G induced by the vertex set V'; in other words, $G[V']$ consists of the vertices V' along with the edges $\gamma_G(V')$. Let $V(G) := V$ and $E(G) := E$. Given a subgraph H of G, let $H + \hat{V}$ be the subgraph H with the vertices $\hat{V} \subset V$ added to it, and let $H + \hat{E}$ be the subgraph H with the edges $\hat{E} \subseteq E$ added to it. For the notation and definitions of polyhedral theory, we refer the reader to Chapter 6 of [5].

2 Complexity of VCC; and the Extended Minimum Cost Spanning Tree Problem

In this section we consider the complexity of the general VCC problem. We show that VCC$(2, 1, V^*)$, and thus VCC(k, l, V^*), is NP-hard, but that there exists a polynomial-time algorithm that solves VCC$(1, 1, V^*)$. We also give a complete linear description for VCC$(1, 1, V^*)$. Both the algorithm and polyhedral description of VCC$(1, 1, V^*)$ will be used in subsequent sections as an essential tool for the approximation algorithm and integrality gap results for the general VCC(k, l, V^*).

Lemma 1 *Problem VCC(*$2, 1, V^*$*) is NP-hard.*

Proof: We show that VCC$(2, 1, V^*)$ is NP-hard by showing that the decision form of multi-2EC is polynomial time reducible to the decision form of VCC$(2, 1, V^*)$. The decision form, D1, of VCC$(2, 1, V^*)$ and the decision form, D2, of multi-2EC are defined as follows:

D1: Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, a set of vital vertices $V^* \subseteq V$, and some $L \in \mathbf{R}_{\geq 0}$, is there a solution of VCC$(2, 1, V^*)$ with total edge cost less than or equal to L?

D2: Given a complete graph $\hat{G} = (\hat{V}, \hat{E})$ with non-negative edge costs $\hat{c} \in \mathbb{R}^{\hat{E}}$, and some $Q \in \mathbf{R}_{\geq 0}$, is there a solution of multi-2EC with total edge cost less than or equal to Q?

Clearly, D1 is in NP. We will complete the proof that VCC$(2, 1, V^*)$ is NP-hard by showing that D2 is polynomially reducible to D1. We create an instance of D1 from an instance of D2 as follows. Let G be the complete graph obtained by adding a non-empty set of vertices V' to $\hat{G}$ and edges such that G is a complete; such that the edge cost of edges in $\gamma_G(V')$ is 0, the edge cost of ru for some vertex $u \in \hat{V}$ and some vertex $r \in V'$ is 0, and all other edges in $\delta_G(V')$ each have edge cost M, where M is twice the total cost of the edges in $\hat{G}$. See Figure 2. Let $V^* = V \backslash V' (= \hat{V})$. Let $L := Q$.

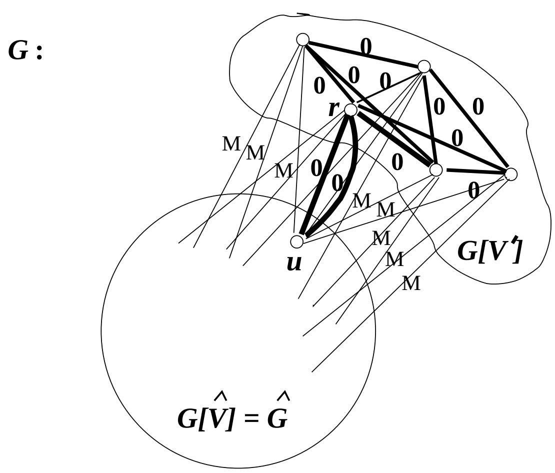


Figure 2: Illustration of G in the proof of Theorem 2. Edges in $\delta_G(V') \backslash \{ru\}$ have cost M.

Suppose H is a solution to VCC$(2, 1, V^*)$ on G with total cost less than or equal to L, i.e., we have a 'yes' instance of problem D1. Due to the edge costs, H has only 2 copies of the edge ru in the cut $\delta_H(V')$ (this satisfies the cut requirement that in any feasible solution there are at least two edges in the cut $\delta_G(V')$, and taking any other edge in $\delta_G(V')$ will make the total cost greater than M). Thus, since H is a multi-two-edge connected subgraph, the multi-subgraphs $H[V']$ and $H[\hat{V}]$ are both multi-2-edge connected subgraphs. Hence, $H[\hat{V}]$ is a feasible solution to multi-2EC on $G[\hat{V}] = \hat{G}$. Furthermore, since the edge costs of ru and the edges in $\gamma_G(V')$ are 0, the total cost of H is equal to the total cost of $H[\hat{V}]$. Thus, the total cost of $H[\hat{V}]$ is less than or equal to Q, and $H[\hat{V}]$ shows that we had a 'yes' instance of problem D2.

Conversely, suppose $\hat{H}$ is a solution to multi-2EC on $\hat{G}$ with total cost less than or equal to Q, i.e., we have a 'yes' instance of problem D2. Clearly, $H' := \hat{H} + V' + \{ru, ru\} + E(G[V'])$ is 2-edge connected and $H'[V^*]$ is connected. Thus, H' is a feasible solution to VCC$(2, 1, V^*)$ on G. The total cost of H' is equal to the total cost of $\hat{H}$. Thus, the total cost of H' is less than or equal to Q, and we had a 'yes' instance of problem D1.

Therefore, the decision form of multi-2EC is polynomial time reducible to the decision form of VCC$(2, 1, V^*)$. Since multi-2EC is an NP-hard problem [3, 12], therefore VCC$(2, 1, V^*)$ is an NP-hard problem. ■

Theorem 2 *Problem VCC(k, l, V^*) is NP-hard.*

Proof: Lemma 1 shows that VCC$(2, 1, V^*)$ is NP-hard. The result then follows from the fact that VCC$(2, 1, V^*)$ is a special case of VCC(k, l, V^*). ■

2.1 A Polynomial-time Algorithm for VCC$(1, 1, V^*)$

We now consider VCC$(1, 1, V^*)$, i.e., VCC when $k = l = 1$: Given a complete graph, $G = (V, E)$, with non-negative edge costs $c \in \mathbb{R}^E$, and a given non-empty subset of vital vertices $V^* \subseteq V$, find a minimum cost spanning tree T of G that remains connected when the secondary vertices $V' = V \backslash V^*$ are removed, i.e., such that $T[V^*]$ is also a spanning tree of the vital core $G[V^*]$. We call this special case of VCC the *Extended Minimum Spanning Tree Problem* (EMST). Instances of EMST are denoted by EMST(V^*). When $V^* = V$, EMST reduces to the minimum cost spanning tree problem (MST), which is the special case VCC$(1, 1, V)$. Thus, we have the following inequality:

Proposition 3 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, and a non-empty subset of vertices $V^* \subseteq V$, the following holds on G: opt$(MST) \leq$ opt$(EMST(V^*))$.*

The following characteristic of a feasible solution of EMST is used by the algorithm for EMST(V^*):

Lemma 4 *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, and let $V^* \subseteq V$ be a non-empty subset of vertices. Let M be an optimal solution to EMST(V^*) on G. Then $M[V^*]$ is a minimum cost spanning tree of $G[V^*]$.*

Proof: Since the removal of the secondary vertices does not disconnect M, therefore, $M[V^*]$ is a connected subgraph of $G[V^*]$. Moreover, since M is a spanning tree of G, therefore $M[V^*]$ is a spanning subgraph of $G[V^*]$ and is acyclic. Since $M[V^*]$ is a connected, acyclic, spanning subgraph of the vital core $G[V^*]$; thus, $M[V^*]$ is a spanning tree of $G[V^*]$. Therefore, $M[V']$ has connected components $T_1, \ldots, T_t$ (for some $t \in \mathbf{N}_{>0}$) which are each trees, and which are each connected in M to $M[V^*]$ by a single distinct edge in $\delta_M(M[V^*])$, say $e_1, \ldots, e_t$ respectively.

Suppose B is a minimum cost spanning tree of $G[V^*]$. Thus, we have that *cost*$(B) \leq$ *cost*$(M[V^*])$. Suppose *cost*$(B) <$ *cost*$(M[V^*])$. Let $\hat{M}$ be the subgraph of G obtained from M by replacing $M[V^*]$ by B; i.e., $\hat{M}$ is the subgraph of

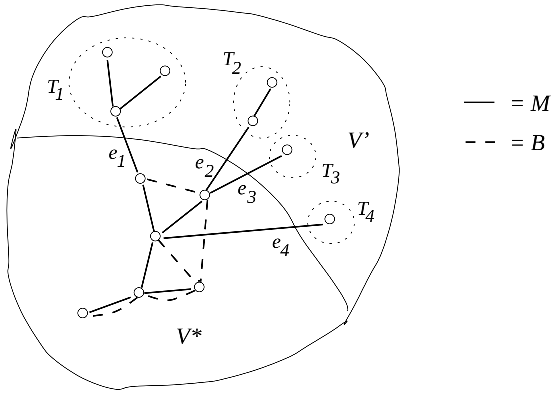


Figure 3: Replacing $M[V^*]$ by B in M.

G consisting exactly of the subtree B, the (disjoint) subtrees $T_1, \ldots, T_t$, and the unique "connecting" edges $e_1, \ldots, e_t$ (see Figure 3). It is easy to see that $\hat{M}$ is a spanning tree of G. Furthermore, $\hat{M}[V^*]$ is connected, since $\hat{M}[V^*]$ is precisely B (which is connected). Thus, $\hat{M}$ is a feasible solution to EMST(V^*) on G, and

$$\begin{aligned} cost(\hat{M}) &= cost(B) + \sum_{i=1}^{t} c_{e_i} + \sum_{i=1}^{t} cost(T_i) \\ &< cost(M[V^*]) + \sum_{i=1}^{t} c_{e_i} + \sum_{i=1}^{t} cost(T_i) \\ &= cost(M), \end{aligned}$$

which is a contradiction with the fact that M is an optimal solution to ENST(V^*) on G. Thus, $cost(B) \not< cost(M[V^*])$. Therefore, $M[V^*]$ is a minimum cost spanning tree of $G[V^*]$. ■

Notice, from Lemma 4, that in a feasible solution of EMST, every secondary vertex $v \in V' = V \backslash V^*$ is either a leaf of the spanning tree, or is part of a branch of the spanning tree such that, from v to the leaf/leaves of the branch, all the vertices are secondary vertices.

Given a graph $G = (V, E)$, and $V^* \subseteq V$, define the multi-graph G/V^* to be the *shrunk graph* of G, obtained by identifying all the vertices in V^* into a single ("shrunk") vertex w and deleting all the edges of G that have both endpoints in V^*, i.e., deleting all the edges $\gamma_G(V^*)$. Notice that G/V^* has $|V| - |V^*| + 1$ vertices, and its edge set, E_{V^*}, consists of all the edges of G that have exactly one or no endpoint in V^*, i.e., $E_{V^*} = E \backslash \gamma_G(V^*) = \delta_G(V^*) \cup \gamma_G(V \backslash V^*)$. Notice that edges iw, $i \in V' = V \backslash V^*$, in G/V^* are in 1 to 1 correspondence with edges $ij \in E$, $j \in V^*$. Edges in G/V^* that are not incident to the shrunk vertex w are in 1 to 1 correspondence with the edges in G that do not have both endpoints in

V^*. Without loss of generality in this paper, we can convert G/V^* into a simple graph by removing all but the lowest cost edge in every set of parallel edges.

Lemma 5 *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, let $V^* \subseteq V$ be a non-empty subset of vertices, and let $V' = V \backslash V^*$. Let M be an optimal solution to EMST(V^*) on G. Then M restricted to the shrunk graph G/V^* is a minimum cost spanning tree of G/V^*.*

Proof: Notice that G/V^* is connected (since G is connected), and thus contains a spanning tree. Let M^* be M restricted to the shrunk graph G/V^*. Since $M[V^*]$ is a spanning tree of $G[V^*]$ (by Lemma 4), therefore, $M[V']$ has connected components $T_1, \ldots, T_t$ (for some $t \in \mathbf{N}_{>0}$) which are each trees, and which are each connected in M to $M[V^*]$ by a single distinct edge in $\delta_M(M[V^*])$, say $e_1, \ldots, e_t$ respectively. Thus, M^* consists of the shrunk vertex w, the edges $e_1, \ldots, e_t$, and the subtrees $T_1, \ldots, T_t$. Therefore, M^* is a spanning tree of G/V^*.

Suppose A is a minimum cost spanning tree of G/V^*. Thus, $cost(A) \leq cost(M^*)$. Suppose $cost(A) < cost(M^*)$. Let $\hat{M}$ be the subgraph of G obtained from M by replacing the edges of M^* by the edges of A; i.e., $\hat{M}$ is the subgraph of G consisting exactly of the subtree $M[V^*]$, the (disjoint) subtrees $A_1, \ldots, A_s$ (for some $s \in \mathbf{N}_{>0}$) of $A - w = A[V']$, and the distinct "connecting" edges $e_1^A, \ldots, e_s^A$, where e_i^A joins the subtree A_i with w in A (for each $i \in \{1, \ldots, s\}$). It is easy to see that $\hat{M}$ is a spanning tree of G. By construction, $cost(\hat{M}) = cost(M[V^*]) + cost(A) < cost(M[V^*]) + cost(M^*) = cost(M)$. This is a contradiction with the assumption that M is a minimum cost spanning tree of G. Thus, $cost(A) = cost(M^*)$, and M^* is a minimum cost spanning tree of G/V^*. ■

Based on Lemmas 4 and 5 above, we have the following polynomial-time algorithm for EMST(V^*):

Algorithm M

Input: A complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, and a non-empty subset of vital vertices $V^* \subseteq V$.

M1. Find a minimum cost spanning tree, $\tilde{M}$, of the vital core $G[V^*]$.

M2. Find a minimum cost spanning tree, M^*, of the shrunk graph G/V^*.

M3. Form the subgraph $M' = \big(V, E(\tilde{M}) \cup E(M^*)\big)$.

Lemma 6 *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, and let $V^* \subseteq V$ be a non-empty subset of vertices. The subgraph M' returned by Algorithm M is an optimal solution to EMST(V^*) on G, and can be constructed in worst case running time $O(|V|^2)$.*

Proof: By Lemmas 4 and 5, an optimal solution to EMST(V^*) on G consists exactly of the vertices V along with the edges of a minimum cost spanning tree of $G[V^*]$ and the edges of a minimum cost spanning tree of G/V^*.[3] This is precisely M'. Thus, M' is an optimal solution to EMST(V^*) on G.

For a connected graph with n vertices and m edges, a minimum cost spanning tree can be found in $O(n \log n + m)$ time [11]. This becomes $O(n^2)$ for a complete graph. Thus, steps (M1) and (M2) can be implemented in worst case running time $O(|V|^2)$. ■

2.2 A Complete Linear Description for VCC$(1, 1, V^*)$

The following is needed in order to provide a complete linear description of EMST(V^*), which is the special case VCC$(1, 1, V^*)$. The description we present will be used in the next section to obtain our approximation algorithm and integrality gap results for VC(k, l, V^*). Given a complete graph $G = (V, E)$, a *partition* $P = (V_1, V_2, \dots, V_{k_P})$ of the vertex set V, is a set of subsets of V such that $V_1 \cup V_2 \cup \cdots \cup V_{k_P} = V$ and $V_i \cap V_j = \emptyset$ for all $i, j \in \{1, 2, \dots, k_P\}$, $i \neq j$. Notice that in such a partition, each *part* V_i induces a complete subgraph $G[V_i]$ for all $i = 1, 2, \dots, k_P$. In this paper, when we refer to a partition P of G, we are referring to a partition P of the vertex set of G. The multi-graph G_P is the graph obtained by identifying all the vertices in V_i, $i = 1, 2, \dots, k_P$, into a single vertex $\tilde{v}_i$ and deleting all the edges of G that have both endpoints in the same part, V_i, of the partition P. Thus, G_P has k_P vertices, and its edge set, E_P, consists of all the edges of G that have endpoints in different parts, V_i, of the partition P. In this paper, without loss of generality, we can convert G_P into a simple graph by removing all but the lowest cost edge in every set of parallel edges. Let x_e represent the number of times each edge $e \in E$ is included in a solution of a given problem on G. The resulting vector $x \in \mathbb{N}^E_{\geq 0}$ is called an *incidence vector* on G. For any edge set $E' \subseteq E$ and $x \in \mathbb{R}^E$, let $x(E')$ denote the sum $\sum_{e \in E'} x_e$. The next lemma follows from results from Chopra [4], as they apply to complete graphs. The lemma states that we can find the cost of a minimum cost spanning tree by solving a linear program that has no integrality constraints. Note that the *spanning tree polytope* is the convex hull of spanning tree incidence vectors.

[3] Notice that, without loss of generality, G/V^* can be made into a simple graph by removing, for each secondary vertex $u \in V'$, all the multiple copies of uw except for the copy of uw with the cheapest edge cost. Thus, any algorithm for finding a minimum cost spanning tree can be used, regardless of whether or not that algorithm starts on a simple or multi-graph.

Lemma 7 ([4], Theorem 2.3) *If $G = (V, E)$ is a complete graph, and $c \in \mathbb{R}^E$ with $c \geq 0$, then the cost of a minimum cost spanning tree of G is equal to the optimal value of*

$$\begin{aligned} \textit{minimize} \quad & cx & & & (1)\\ \textit{subject to} \quad \sum_{e \in E_P} & x_e & \geq & \; k_P - 1, \quad \textit{for all partitions } P \textit{ of } G, & (2)\\ & x_e & \geq & \; 0, \quad \textit{for all } e \in E. & (3) \end{aligned}$$

Moreover, the feasible region of LP(1) is precisely the dominant of the spanning tree polytope.

Consider the following linear programming (LP) formulation on a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and a given subset of vertices $V^* \subseteq V$. We will show that this is a linear programming formulation for EMST(V^*):

$$\begin{aligned} \text{minimize} \quad & cx & & & (4)\\ \text{subject to} \quad \sum_{e \in E_P} & x_e & \geq & \; k_P - 1, \; \text{for all partitions } P \text{ of } G/V^*, & (5)\\ \sum_{e \in E_P} & x_e & \geq & \; k_P - 1, \; \text{for all partitions } P \text{ of } G[V^*], & (6)\\ & x_e & \geq & \; 0, \quad \text{for all } e \in E. & (7) \end{aligned}$$

Notice that constraints (5) and (7) are precisely those constraints for the minimum spanning tree LP(1) on the shrunk graph G/V^* of G, and constraints (6) and (7) are precisely those constraints for the minimum spanning tree LP(1) on the vital core $G[V^*]$.

We will now show that LP(4) is a linear programming formulation for EMST(V^*) whose integrality gap is 1; i.e., for which there always exists an optimal solution that is integer. Let $P_{EMST(V^*)}$ be the associated polytope of EMST(V^*), i.e., let $P_{EMST(V^*)}$ be the convex hull of the incidence vectors of solutions of EMST(V^*). Let DP be the dominant[4] of $P_{EMST(V^*)}$. Let P_{MST} be the associated polytope of the minimum cost spanning tree problem. By Lemma 7, the dominant of P_{MST} is given by the feasible region of LP(1).

Theorem 8 *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, and let $V^* \subseteq V$. The dominant, DP, of $P_{EMST(V^*)}$ is given by the feasible region of LP(4).*

[4]The *dominant* of a polytope P is the polyhedron formed by P plus the non-negative orthant $\mathbb{R}^n_+$).

Proof: We show that every feasible solution to the constraints (5)-(7) can be expressed as a convex combination of incidence vectors of optimal solutions of EMST(V^*), i.e., extreme points of DP, plus some non-negative vector $h \in \mathbb{R}^E_{\geq 0}$.

Let x be a feasible solution to the constraints (5)-(7), i.e., x is a point in the feasible region determined by the constraints (5)-(7). Let x_1 be x restricted to the edges in $G[V^*]$, and let x_2 be x restricted to the edges in G/V^*. Thus, by Lemma 7, x_1 and x_2 are feasible points in the dominant of P_{MST} on $G[V^*]$ and G/V^*, respectively. Thus, x_1 can be written as a convex combination of the extreme points of the dominant of P_{MST} on $G[V^*]$, plus a non-negative vector $f \in \mathbb{R}^{\gamma(V^*)}_{\geq 0}$; i.e., let $x_1 = \lambda_1 y_1 + \lambda_2 y_2 + \ldots + \lambda_t y_t + f$ where y_i is an extreme point of the dominant of the MST polytope on $G[V^*]$ (i.e., y_i is the incidence vector of an MST of $G[V^*]$), and λ_i is a non-negative multiplier such that $\lambda_1 + \cdots + \lambda_t = 1$, for $i \in \{1, 2, \ldots, t\}$. Similarly, let $x_2 = \alpha_1 z_1 + \alpha_2 z_2 + \ldots + \alpha_s z_s + g$ where $g \in \mathbb{R}^{E_{V^*}}_{\geq 0}$ is a non-negative vector, z_j is an extreme point of the dominant of the MST polytope on G/V^* (i.e., z_j is the incidence vector of an MST of G/V^*), and α_j is a non-negative multiplier such that $\alpha_1 + \cdots + \alpha_s = 1$, for $j \in \{1, 2, \ldots, s\}$.

By Lemmas 4 and 5 applied to the corresponding incidence vectors, any y_i, $i \in \{1, 2, \ldots, t\}$, combined with any z_j, $j \in \{1, 2, \ldots, s\}$, gives an extreme point of DP. Let β be the lowest common denominator of $\lambda_1, \ldots, \lambda_t, \alpha_1, \ldots, \alpha_s$, and rewrite each λ_i and each α_j as a sum of β. Expanding these out in the sums of x_1 and x_2, we then get that x_1 and x_2 are both a sum consisting of β terms (plus f, respectively g), with each term having the multiplier $1/\beta$. Therefore, combining together each 'corresponding' k^{th} term of x_1 and x_2, $k = 1, 2, \ldots, \beta$, and letting $h \in \mathbb{R}^E_{\geq 0}$ be the vector by combining f and g, we obtain $x - h$ written as a linear combination of extreme points of DP. Moreover, this combination has exactly β terms, each with a multiplier of $1/\beta$, and thus the multipliers add up to 1. Therefore, x can be written as a convex combination of extreme points of DP plus a non-negative vector in $\mathbb{R}^E_{\geq 0}$. Thus, feasible region determined by the constraints (5)-(7) is the dominant of $P_{EMST(V^*)}$. ■

Notice that Theorem 8 can also be proved using matriods.

Corollary 9 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, and a non-empty subset of vertices $V^* \subseteq V$, there exists an optimal solution to LP(4) that is the incidence vector of a minimum cost spanning tree of G. Thus, LP(4) is a linear programming formulation for EMST(V^*) whose integrality gap is 1. In particular, the cost of an optimal solution to EMST(V^*) is equal to the optimal value of LP(4).*

Proof: Since we are minimizing over DP, and, from Theorem 8, all the extreme points of DP are incidence vectors of optimal solutions of EMST(V^*), therefore there exists an optimal solution to LP (4) that is the incidence vector of a minimum cost spanning tree of G. ■

3 The Vital Core Connectivity Problem

In this section we present our integrality gap and approximation algorithm results for VCC, and the preliminary results necessary in order to obtain them. Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, let $V^* \subseteq V$ be a non-empty set of vital vertices, $V' = V \backslash V^*$ be the secondary vertices, and let $k, l \in \mathbb{Z}$, $1 \leq l \leq k$ be non-negative integers. An integer linear programming formulation for VCC(k, l, V^*) is as follows:

$$\begin{aligned} \text{minimize} \quad & cx & & & (8)\\ \text{subject to} \quad x(\delta_G(S)) & \geq k, && \text{for all } \emptyset \subset S \subset V, & (9)\\ x(\delta_{G[V^*]}(S)) & \geq l, && \text{for all } \emptyset \subset S \subset V^*, & (10)\\ x_e & \geq 0, && \text{for all } e \in E, & (11)\\ x_e & && \text{integer, for all } e \in E. & (12) \end{aligned}$$

The constraints (9) ensure that the multi-subgraph is k-edge connected; and the constraints (10) ensure that the multi-subgraph on the vital core V^* is l-edge connected. The linear programming formulation of the LP relaxation of VCC(k, l, V^*) (denoted by VCC$(k, l, V^*)_{LP}$) is ILP(8) without the integer constraints (12), and will be referred to as LP(VCC). The linear programming formulation of the LP relaxation of multi-kEC on G (denoted by multi-kEC$_{LP}$) is ILP(8) without the constraints (10) and (12).

Proposition 10 *On a complete graph $G = (V, E)$, problem VCC(k, l, V) is equivalent to multi-kEC, and their LP relaxations are also equivalent. Thus, their optimal values are equal, as are the optimal values of their LP relaxations.*

Proof: When $V^* = V$, the constraints (10) are: $x(\delta_G(S)) \geq l$, for all $\emptyset \subset S \subset V$. Since $k \geq l$, these constraints are redundant, and thus LP(VCC) consists of just the constraints (9) and (11), which is precisely the LP of multi-kEC$_{LP}$. Clearly, the same applies for the respective ILP formulation. The results follow. ■

Notice that, from the proof of Proposition 10, we can, without loss of generality, set $l = k$ for instances of VCC in which $V^* = V$.

3.1 Preliminary Results

Before proceeding to our approximation algorithm and an upper bound on the integrality gap of the LP relaxation of VCC(k, l, V^*), a few general results are needed. We briefly present these results now.

Given a connected graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, and given $T \subseteq V$, a *minimum cost T-join* is a minimum cost set of edges $\tilde{E} \subseteq E$ such that $|\delta(v) \cap \tilde{E}|$ is odd if and only if $v \in T$. The following theorem is needed:

Theorem 11 (Cook et. al. [5], Theorem 5.28) *If $G = (V, E)$, $T \subseteq V$ with $|T|$ even, and $c \in \mathbb{R}^E$ with $c \geq 0$, then the minimum cost of a T-join of G is equal to the optimal value of*

$$\begin{aligned} & \textit{minimize} \;\; cx & & & \quad (13) \\ \textit{subject to} \quad & x(\delta(S)) \;\geq\; 1, & & \textit{for all } S \subseteq V \textit{ s.t.} & \\ & & & |S \cap T| \textit{ odd}, & \quad (14) \\ & x_e \;\geq\; 0, & & \textit{for all } e \in E. & \quad (15) \end{aligned}$$

Lemma 12 *Let $T \subseteq V$, $|T|$ even. The following holds on G:*

$$opt_G(T\text{-join}) \;\leq\; \frac{1}{k}\, opt_G(\mathrm{VCC}(k, l, V^*)_{LP}) \;\leq\; \frac{1}{k}\, opt_G(\mathrm{VCC}(k, l, V^*)).$$

Proof: Let x^* be an optimal solution to LP(VCC). In particular, x^* satisfies the constraints (9). Thus, $\frac{1}{k}x^*$ satisfies $x(\delta_G(S)) \geq 1$ for all $\emptyset \subset S \subset V$. Therefore $\frac{l}{k}x^*$ satisfies constraints (14). Additionally, since x^* satisfies the constraints (11), $\frac{1}{k}x^*$ also satisfies the constraints (15). Therefore, $\frac{1}{k}x^*$ satisfies the constraints (14) and (15), and thus is a feasible solution of LP(13). Using Theorem 11, we have that

$$opt(\textit{T-join}) \leq \frac{1}{k}\; cx^* \;= \frac{1}{k}\; opt(\textit{VCC}(k, l, V^*)_{LP}) \leq \frac{1}{k}\; opt(\textit{VCC}(k, l, V^*)).$$

■

Corollary 13 *Let $\tilde{T} \subseteq V^*$, $|\tilde{T}|$ even. The following relationship exists on G and $G[V^*]$:*

$$opt_{G[V^*]}(\tilde{T}\text{-join}) \;\leq\; \frac{1}{l}\; opt_G(\mathrm{VCC}(k, l, V^*)_{LP}) \;\leq\; \frac{1}{l}\; opt_G(\mathrm{VCC}(k, l, V^*)).$$

Proof: By Lemma 12, $opt_{G[V^*]}(\tilde{T}\textit{-join}) \leq \frac{1}{l}\; opt_{G[V^*]}(\textit{VCC}(l, l, V^*)_{LP})$. Clearly, $opt_{G[V^*]}(\textit{VCC}(l, l, V^*)_{LP}) \leq opt_G(\textit{VCC}(k, l, V^*)_{LP})$. The result follows. ■

The next lemma gives an upper bound on EMST(V^*).

Lemma 14 *The following holds on G:*

$$opt_G(\mathrm{EMST}(V^*)) \leq \frac{2}{l}\; opt_G(\mathrm{VCC}(k, l, V^*)_{LP}) \leq \frac{2}{l}\; opt_G(\mathrm{VCC}(k, l, V^*)).$$

Proof: Let x^* be an optimal solution to LP(VCC) on G and let $x' = \frac{2}{l}x^*$. Since x^* satisfies the constraints (9) and (10), and since $k \geq l$, therefore x' satisfies

$$x'(\delta(S)) \;\geq\; \frac{2k}{l} \;\geq\; 2, \qquad\qquad \text{for all } \emptyset \subset S \subset V \qquad (16)$$

and

$$x'(\delta_{G[V^*]}(S)) \;\geq\; \frac{2l}{l} \;=\; 2, \qquad\qquad \text{for all } \emptyset \subset S \subset V^*. \tag{17}$$

Let $P = (S_1, S_2, \ldots, S_{k_P})$ be a partition of G. For x' on G we have, using (16):

$$\begin{aligned}
\sum_{e \in E_P} x'_e \;&=\; \frac{1}{2}\big[x'_G(\delta(S_1)) + x'_G(\delta(S_2)) + \ldots + x'_G(\delta(S_{k_P}))\big] \\
&\geq\; \frac{1}{2}\,(k_P \cdot 2) \;=\; k_P \\
&\geq\; k_P - 1.
\end{aligned}$$

Therefore, x' satisfies $\sum_{e \in E_P} x'_e \geq k_P - 1$, for all partitions $P = (S_1, S_2, \ldots, S_{k_P})$ of G. Thus, x' satisfies the constraints (5).

Similarly, let $\tilde{P} = (T_1, T_2, \ldots, T_{k_{\tilde{P}}})$ be a partition of $G[V^*]$. For x' on $G[V^*]$ we have, using (17):

$$\begin{aligned}
\sum_{e \in E_{\tilde{P}}} x'_e \;&=\; \frac{1}{2}\big[x'_{G[V^*]}(\delta(T_1)) + x'_{G[V^*]}(\delta(T_2)) + \ldots + x'_{G[V^*]}(\delta(T_{k_{\tilde{P}}}))\big] \\
&\geq\; \frac{1}{2}\,(k_{\tilde{P}} \cdot 2) \;=\; k_{\tilde{P}} \\
&\geq\; k_{\tilde{P}} - 1.
\end{aligned}$$

Thus, x' satisfies $\sum_{e \in E_{\tilde{P}}} x'_e \geq k_{\tilde{P}} - 1$, for all partitions $\tilde{P}$ of $G[V^*]$, and therefore, x' satisfies the constraints (6).

Clearly, x' satisfies the constraints (7). Thus, $x' = \frac{2}{l}x^*$ is a feasible solution of LP(4), and we therefore have that $opt(EMST(V^*)) \leq \frac{2}{l}\, opt(VCC(k,l,V^*)_{LP})$. Noticing that $opt(VCC(k,l,V^*)_{LP}) \leq opt(VCC(k,l,V^*))$ completes the proof. ■

Corollary 15 *The following holds on G:*

$$opt_G(\text{MST}) \leq \frac{2}{k}\, opt_G(\text{VCC}(k,l,V^*)_{LP}) \leq \frac{2}{k}\, opt_G(\text{VCC}(k,l,V^*)).$$

Proof: Notice that MST is precisely the special case EMST(V). Thus, from Lemma 14, $opt(MST) \leq \frac{2}{l}\, opt(VCC(k,l,V)_{LP}) = \frac{2}{k}\, opt(VCC(k,l,V)_{LP})$. By Proposition 10, $opt(VCC(k,l,V)_{LP}) \leq opt(VCC(k,l,V^*)_{LP})$. The result follows. ■

3.2 Vital Core Connectivity – Approximation Algorithm and Integrality Gap

Lemmas 12 and 14, and Corollaries 15 and 13 will be used to create a solution for VCC(k, l, V^*) within a given bound of the optimal. Before presenting these upper bounds, and the special cases in which the upper bounds can be further improved, we present an approximation algorithm for VCC(k, l, V^*). This algorithm forms a feasible solution for VCC(k, l, V^*) by combining together copies of the following: An extended minimum cost spanning tree of G with respect to V^*, a T-join on G, a $\tilde{T}$-join on $G[V^*]$, and a minimum cost spanning tree of G. Using appropriate combinations of these, we can get a multi-subgraph H with the desired edge connectivity on H and on $H[V^*]$. Notice that for $l = 1$, the approximation algorithm and its upper bound, as well as the upper bound for the integrality gap, is in the Special Cases section (Section 3.3).

Algorithm VCC

Input: A complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, a non-empty subset of vital vertices $V^* \subseteq V$, and non-negative integers $k, l \in \mathbb{Z}$, $2 \leq l \leq k$.

1. Using Algorithm M from Section 2.1, find an extended minimum cost spanning tree, M', of G with vital vertices V^*. Let $E' \subset E$ be the edge set of M'.

2. Let $\hat{V}^{odd} \subset V$ be the set of vertices of G having odd degree in M'. Find a minimum cost $\hat{T}$-join $\hat{J} \subseteq E$, on G, where $\hat{T} = \hat{V}^{odd}$, i.e., M' has a corresponding $\hat{V}^{odd}$-join. This is accomplished by finding a minimum cost "pairing" of the odd degree vertices of M' using minimum cost paths from G.

3. Let $\tilde{M} := M'[V^*]$, i.e., $\tilde{M}$ is the minimum spanning spanning tree of $G[V^*]$ obtained by restricting M' to $G[V^*]$ (cf Lemma 4). Let $\tilde{V}^{odd} \subset V$ be the set of vertices of G having odd degree in $\tilde{M}$. Find a minimum cost $\tilde{T}$-join $\tilde{J} \subseteq E$, on $G[V^*]$, where $\tilde{T} = \tilde{V}^{odd}$.

4a. If $l \geq \lceil \frac{k}{2} \rceil$:
Take $\lceil \frac{k}{2} \rceil$ copies of the edges E', and let this form the edge set E'';
Take $\lfloor \frac{k}{2} \rfloor$ copies of the edges in $\hat{J}$, and let this form the edge set $\hat{J}'$;
Take $\left(l - \lceil \frac{k}{2} \rceil\right)$ copies of the edges in $\tilde{J}$, and let this form the edge set $\tilde{J}'$.
Combine these copies of the spanning tree M', of the $\tilde{T}$-join, and of the $\hat{T}$-join, i.e.: Form the multi-subgraph $H := (V, E'' \cup \hat{J}' \cup \tilde{J}')$.

4b. If $l < \lceil \frac{k}{2} \rceil$:
Take $\lceil \frac{l}{2} \rceil$ copies of the edges E', and let this form the edge set $\hat{E}''$;
Take $\lfloor \frac{l}{2} \rfloor$ copies of the edges in $\tilde{J}$, and let this form the edge set $\tilde{J}''$;
Take $\lfloor \frac{l}{2} \rfloor$ copies of the edges in $\hat{J}$, and let this form the edge set $\hat{J}''$.
Find a minimum cost spanning tree, M, of G. Let $E_M \subset E$ be the edge set of M. Take $\lceil \frac{k-l}{2} \rceil$ copies of its edges, and let this form the edge set E'_M.
Let $V^{odd} \subset V$ be the set of vertices of G having odd degree in M. Find a minimum cost T-join $J \subseteq E$, on G, where $T = V^{odd}$. Take $\lfloor \frac{k-l}{2} \rfloor$ copies of the edges in J, and let this form the edge set J'.
Combine these copies of the spanning tree M', of the $\tilde{T}$-join, of the spanning tree M, and of the T-join, i.e.: Form the multi-subgraph $H := (V, \hat{E}'' \cup \tilde{J}'' \cup \hat{J}'' \cup E'_M \cup J')$.

5. Return H.

Proposition 16 *Algorithm VCC runs in polynomial time and produces a feasible solution of VCC(k, l, V^*).*

Proof: A minimum cost spanning tree can be found in $O(|V| \log |V| + |E|)$ time [11]. This becomes $O(|V|^2)$ for a complete graph. Finding a minimum cost T-join can be done in the worst case running time $O(|V|^3)$ [3]. Therefore, using Lemma 6, Algorithm VCC has worst case running time $O(|V|^3+|V|^3+|V|^3+|V|^2+|V|^2) = O(|V|^3)$.

We now show that the graph H returned by Algorithm VCC is a feasible solution to VCC(k, l, V^*). We show first that H is a k-edge connected spanning multi-subgraph of G. Since a connected graph has a T-join if and only if $|T|$ is even, and every graph has an even number of odd degree vertices, therefore, G has a $\hat{V}^{odd}$-join and a V^{odd}-join, and $G[V^*]$ has a $\tilde{V}^{odd}$-join. Clearly, all the vertices of the multi-subgraph $K' := (V, E' \cup \hat{J})$ have even degree. This means that K' is an Eulerian graph, i.e., it contains an Euler tour. Hence, K' is two-edge connected, i.e., it has two edge disjoint paths between every pair of vertices. Similarly, the multi-subgraph $K := (V, E_M \cup J)$ is two-edge connected.

Suppose $l \geq \lceil \frac{k}{2} \rceil$. When k is even, the edges of H partition into $\frac{k}{2}$ extended spanning trees and $\frac{k}{2}$ $\hat{V}^{odd}$-joins (in particular, into $\frac{k}{2}$ copies of K'). Hence, when k is even, H partitions into $\frac{k}{2}$ Eulerian subgraphs, i.e., $\frac{k}{2}$ 2-edge connected subgraphs. Therefore, H is $(\frac{k}{2} \cdot 2) = k$-edge connected. When k is odd, the edges of H partition into $\frac{k+1}{2}$ extended spanning trees and $\frac{k-1}{2}$ $\hat{V}^{odd}$-joins; thus, H partitions into $\frac{k-1}{2}$ Eulerian subgraphs and one (extended) spanning tree of G, i.e., $\frac{k-1}{2}$ 2-edge connected subgraphs and one connected subgraph. Thus, H has $(\frac{k-1}{2} \cdot 2 + 1) = k$ edge-disjoint paths (counting multiple copies of edges as distinct edges) between any two vertices, and is therefore k-edge connected. Hence, H

is a k-edge connected spanning multi-subgraph of G. Next, consider the edge connectivity of $H[V^*]$. Notice that the edges of $H[V^*]$ partition into $\lceil \frac{k}{2} \rceil$ copies of the edges of $\tilde{M}$, and $l - \lceil \frac{k}{2} \rceil$ copies of the edges in $\tilde{J}$. Thus, $H[V^*]$ partitions into $l - \lceil \frac{k}{2} \rceil$ Eulerian subgraphs of $G[V^*]$ and $\big(\lceil \frac{k}{2} \rceil - (l - \lceil \frac{k}{2} \rceil)\big) = 2\lceil \frac{k}{2} \rceil - l$ spanning trees of $H[V^*]$, i.e., $\big(l - \lceil \frac{k}{2} \rceil\big)$ 2-edge connected subgraphs and $\big(2\lceil \frac{k}{2} \rceil - l\big)$ connected subgraphs. Thus, $H[V^*]$ has $\Big(2\big(l - \lceil \frac{k}{2} \rceil\big) + \big(2\lceil \frac{k}{2} \rceil - l\big)\Big) = l$ edge-disjoint paths between any two vertices, and is therefore l-edge connected. Hence, $H[V^*]$ is a l-edge connected spanning multi-subgraph of $G[V^*]$.

On the other hand, suppose that $l < \lceil \frac{k}{2} \rceil$. The edges of $H[V^*]$ partition into $\lceil \frac{l}{2} \rceil$ copies of the edges of $\tilde{M}$, and $\lfloor \frac{l}{2} \rfloor$ copies of the edges in $\tilde{J}$. Thus, $H[V^*]$ partitions into $\lfloor \frac{l}{2} \rfloor$ Eulerian subgraphs of $G[V^*]$ and $\big(\lceil \frac{l}{2} \rceil - \lfloor \frac{l}{2} \rfloor\big)$ spanning trees of $H[V^*]$, i.e., $\lfloor \frac{l}{2} \rfloor$ 2-edge connected subgraphs and $\big(\lceil \frac{l}{2} \rceil - \lfloor \frac{l}{2} \rfloor\big)$ connected subgraphs. Thus, $H[V^*]$ has $\Big(2\lfloor \frac{l}{2} \rfloor + \big(\lceil \frac{l}{2} \rceil - \lfloor \frac{l}{2} \rfloor\big)\Big) = l$ edge-disjoint paths between any two vertices, and is therefore l-edge connected. Hence, $H[V^*]$ is a l-edge connected spanning multi-subgraph of $G[V^*]$. Next, the edges of H partition into $\lceil \frac{l}{2} \rceil$ copies of the edges of M', $\lfloor \frac{l}{2} \rfloor$ copies of the edges in $\hat{J}$, $\lceil \frac{k-l}{2} \rceil$ copies of the edges of M, and $\lfloor \frac{k-l}{2} \rfloor$ copies of the edges in J. Thus, the edges of H partition into an l-edge connected subgraph of G and a $(k-l)$-edge connected subgraph of G. Hence, H is $(l + k - l) = k$-edge connected.

The result follows. ■

We will use the following value repeatedly throughout the remainder of this paper: Given non-negative integers $k, l \in \mathbb{Z}$ with $2 \leq l \leq k$, define $\omega_{k,l}$ to be:

- $\frac{1}{2}\big(3 + \frac{k}{l}\big)$, when $l \geq \lceil \frac{k}{2} \rceil$ and k is even;
- $\frac{1}{2}\Big[\big(3 + \frac{k}{l}\big) + \big(\frac{1}{l} - \frac{1}{k}\big)\Big]$, when $l \geq \lceil \frac{k}{2} \rceil$ and k is odd;
- $\big(3 - \frac{l}{k}\big)$, when $l < \lceil \frac{k}{2} \rceil$ and k, l even;
- $\big(3 - \frac{l}{k}\big) + \frac{1}{2l}$, when $l < \lceil \frac{k}{2} \rceil$ and l odd, k even;
- $\big(3 - \frac{l}{k}\big) + \frac{1}{2k}$, when $l < \lceil \frac{k}{2} \rceil$ and l even, k odd;
- $\big(3 - \frac{l}{k}\big) + \big(\frac{1}{2l} - \frac{1}{2k}\big)$, when $l < \lceil \frac{k}{2} \rceil$ and k, l odd.

Notice that $\omega_{k,l} \leq 2\frac{1}{2}$ when $l \geq \lceil \frac{k}{2} \rceil$ and k is even; $\omega_{k,l} \leq \frac{5}{2} + \frac{1}{6} = 2\frac{2}{3}$ when $l \geq \lceil \frac{k}{2} \rceil$ and k is odd; $\omega_{k,l} < 3$ when $2 \leq l < \lceil \frac{k}{2} \rceil$ and k, l even; and $\omega_{k,l} < 3\frac{1}{6}$ when $2 \leq l < \lceil \frac{k}{2} \rceil$ and l odd, k even or l even, k odd or k, l odd. Thus, in general, $\omega_{k,l} \leq 2\frac{2}{3} = \frac{8}{3}$ when $l \geq \lceil \frac{k}{2} \rceil$ and $\omega_{k,l} < 3\frac{1}{6} = \frac{19}{6}$ when $2 \leq l < \lceil \frac{k}{2} \rceil$.

Theorem 17 (Upper Bound of Integrality Gap of VCC(k, l, V^*)$_{LP}$) *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, let $V^* \subseteq V$ be a non-empty set of vital vertices, and let $k, l \in \mathbb{Z}$, $2 \leq l \leq k$ be non-negative integers. The integrality gap for the linear programming relaxation of VCC(k, l, V^*) is at most $\omega_{k,l}$.*

Proof: Let H be the feasible solution of VCC(k, l, V^*) that is returned by Algorithm VCC. *Case 1:* Suppose $l \geq \lceil \frac{k}{2} \rceil$. Then, using Lemma 14, Corollary 13, and Lemma 12, we have,

$$\begin{aligned}
cost(H) &= \left\lceil \frac{k}{2} \right\rceil \cdot opt_G(EMST(V^*)) + \Big(l - \left\lceil \frac{k}{2} \right\rceil\Big) \cdot opt_{G[V^*]}(\tilde{T}\text{-}join) \\
&\qquad + \left\lfloor \frac{k}{2} \right\rfloor \cdot opt_G(\hat{T}\text{-}join) \\
&\leq \left(\left\lceil \frac{k}{2} \right\rceil \cdot \frac{2}{l} \; + \; \Big(l - \left\lceil \frac{k}{2} \right\rceil\Big) \cdot \frac{1}{l} \; + \; \left\lfloor \frac{k}{2} \right\rfloor \cdot \frac{1}{k} \right) \\
&\qquad \cdot opt_G(VCC(k, l, V^*)_{LP}). \qquad (18)
\end{aligned}$$

Case 1a: If k is even, (18) becomes

$$\begin{aligned}
cost(H) &\leq \left(\frac{k}{2} \cdot \frac{2}{l} \; + \; \Big(l - \frac{k}{2}\Big) \cdot \frac{1}{l} \; + \; \frac{k}{2} \cdot \frac{1}{k} \right) \cdot opt_G(VCC(k, l, V^*)_{LP}) \\
&= \left(\frac{k}{l} + 1 - \frac{k}{2l} + \frac{1}{2} \right) \cdot opt_G(VCC(k, l, V^*)_{LP}) \\
&= \frac{1}{2}\Big(3 + \frac{k}{l}\Big) \cdot opt_G(VCC(k, l, V^*)_{LP}). \qquad (19)
\end{aligned}$$

Case 1b: If k is odd, (18) becomes

$$\begin{aligned}
cost(H) &\leq \left(\Big(\frac{k+1}{2}\Big) \cdot \frac{2}{l} \; + \; \Big(l - \frac{k+1}{2}\Big) \cdot \frac{1}{l} \; + \; \Big(\frac{k-1}{2}\Big) \cdot \frac{1}{k} \right) \\
&\qquad \cdot opt_G(VCC(k, l, V^*)_{LP}) \\
&= \left(\frac{2k}{2l} + \frac{2}{2l} + 1 - \frac{k}{2l} - \frac{1}{2l} + \frac{1}{2} - \frac{1}{2k} \right) \cdot opt_G(VCC(k, l, V^*)_{LP}) \\
&= \frac{1}{2}\left(\Big(3 + \frac{k}{l}\Big) + \Big(\frac{1}{l} - \frac{1}{k}\Big) \right) \cdot opt_G(VCC(k, l, V^*)_{LP}). \qquad (20)
\end{aligned}$$

Case 2: Suppose $l < \lceil \frac{k}{2} \rceil$. Then, using Lemmas 12 and 14, and Corollaries 15 and 13, we have,

$$\begin{aligned}
cost(H) &= \left\lceil \frac{l}{2} \right\rceil \cdot opt_G(EMST(V^*)) + \left\lfloor \frac{l}{2} \right\rfloor \cdot opt_{G[V^*]}(\tilde{T}\text{-}join) + \left\lfloor \frac{l}{2} \right\rfloor \cdot opt_G(\hat{T}\text{-}join) \\
&\quad + \left\lfloor \frac{k-l}{2} \right\rfloor \cdot opt_G(T\text{-}join) + \left\lceil \frac{k-l}{2} \right\rceil \cdot opt_G(MST) \\
&\leq \left(\left\lceil \frac{l}{2} \right\rceil \cdot \frac{2}{l} + \left\lfloor \frac{l}{2} \right\rfloor \cdot \frac{1}{l} + \left(\left\lfloor \frac{l}{2} \right\rfloor + \left\lfloor \frac{k-l}{2} \right\rfloor \right) \cdot \frac{1}{k} + \left\lceil \frac{k-l}{2} \right\rceil \cdot \frac{2}{k} \right) \\
&\qquad \cdot opt_G(VCC(k,l,V^*)_{LP}). \qquad (21)
\end{aligned}$$

Case 2a: When k, l are even, (21) becomes

$$\begin{aligned}
cost(H) &\leq \left(\frac{l}{2} \cdot \frac{2}{l} \ + \ \frac{l}{2} \cdot \frac{1}{l} \ + \ \left(\frac{l}{2} + \frac{k-l}{2} \right) \cdot \frac{1}{k} \ + \ \left(\frac{k-l}{2} \right) \cdot \frac{2}{k} \right) \\
&\qquad \cdot opt_G(VCC(k,l,V^*)_{LP}) \\
&= \left(1 + \frac{1}{2} + \frac{l}{2k} + \frac{1}{2} - \frac{l}{2k} + 1 - \frac{l}{k} \right) \cdot opt_G(VCC(k,l,V^*)_{LP}) \\
&= \left(3 - \frac{l}{k} \right) \cdot opt_G(VCC(k,l,V^*)_{LP}). \qquad (22)
\end{aligned}$$

Case 2b: When l is odd and k is even, (21) becomes

$$\begin{aligned}
cost(H) &\leq \left(\left(\frac{l+1}{2} \right) \cdot \frac{2}{l} \ + \ \left(\frac{l-1}{2} \right) \cdot \frac{1}{l} \ + \ \left(\frac{l-1}{2} + \frac{k-l-1}{2} \right) \cdot \frac{1}{k} \right. \\
&\qquad \left. + \left(\frac{k-l+1}{2} \right) \cdot \frac{2}{k} \right) \cdot opt_G(VCC(k,l,V^*)_{LP}) \\
&= \left(1 + \frac{1}{l} + \frac{1}{2} - \frac{l}{2l} + \frac{l}{2k} - \frac{1}{2k} + \frac{1}{2} - \frac{l}{2k} - \frac{1}{2k} + 1 - \frac{l}{k} + \frac{1}{k} \right) \\
&\qquad \cdot opt_G(VCC(k,l,V^*)_{LP}) \\
&= \left(\left(3 - \frac{l}{k} \right) + \frac{1}{2l} \right) \cdot opt_G(VCC(k,l,V^*)_{LP}). \qquad (23)
\end{aligned}$$

Case 2c: When l is even and k is odd, (21) becomes

$$\begin{aligned} cost(H) &\leq \left(\frac{l}{2} \cdot \frac{2}{l} + \frac{l}{2} \cdot \frac{1}{l} + \left(\frac{l}{2} + \frac{k-l-1}{2}\right) \cdot \frac{1}{k} \right. \\ &\qquad \left. + \left(\frac{k-l+1}{2}\right) \cdot \frac{2}{k}\right) \cdot opt_G(VCC(k, l, V^*)_{LP}) \\ &= \left(\left(3 - \frac{l}{k}\right) + \frac{1}{2k}\right) \cdot opt_G(VCC(k, l, V^*)_{LP}). \end{aligned} \tag{24}$$

Case 2d: When k, l are odd, (21) becomes

$$\begin{aligned} cost(H) &\leq \left(\left(\frac{l+1}{2}\right) \cdot \frac{2}{l} + \left(\frac{l-1}{2}\right) \cdot \frac{1}{l} + \left(\frac{l-1}{2} + \frac{k-l}{2}\right) \cdot \frac{1}{k} \right. \\ &\qquad \left. + \left(\frac{k-l}{2}\right) \cdot \frac{2}{k}\right) \cdot opt_G(VCC(k, l, V^*)_{LP}) \\ &= \left(\left(3 - \frac{l}{k}\right) + \frac{1}{2l} - \frac{1}{2k}\right) \cdot opt_G(VCC(k, l, V^*)_{LP}). \end{aligned} \tag{25}$$

Therefore,

$$opt_G(VCC(k, l, V^*)) \leq cost(H) \leq \omega_{k,l} \cdot opt_G(VCC(k, l, V^*)_{LP}),$$

i.e.,

$$\frac{opt_G(VCC(k, l, V^*))}{opt_G(VCC(k, l, V^*)_{LP})} \leq \omega_{k,l}, \tag{26}$$

where $\omega_{k,l}$ is the bound from equation (19), (20), (22), (23), (24), or (25), depending on whether we are in Case 1a, 1b, 2a, 2b, 2c, or 2d, respectively. The result follows. ■

The next corollary follows directly from Theorem 17 and its proof.

Corollary 18 *Let H be the feasible solution of $VCC(k, l, V^*)$ that is returned by Algorithm VCC on G. Then, on G,*

$$cost(H) \leq \omega_{k,l} \cdot opt(\text{VCC}(k, l, V^*)_{LP}) \leq \omega_{k,l} \cdot opt(\text{VCC}(k, l, V^*)).$$

Proof: The first inequality follows directly from equations (19), (20), (22), and (23). Noting that $opt(VCC(k, l, V^*)_{LP}) \leq opt(VCC(k, l, V^*))$ completes the proof. ■

Corollary 19 (Approximation Guarantee of Algorithm VCC) *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, let $V^* \subseteq V$ be a non-empty set of vital vertices, and let $k, l \in \mathbb{Z}$, $2 \leq l \leq k$ be non-negative integers. Algorithm VCC is an $\omega_{k,l}$-approximation algorithm for VCC(k, l, V^*). In particular, Algorithm VCC is an $\frac{8}{3}$-approximation algorithm when $l \geq \lceil \frac{k}{2} \rceil$ and a $\frac{19}{6}$-approximation algorithm when $2 \leq l < \lceil \frac{k}{2} \rceil$.*

Proof: Follows directly from Proposition 16 and Corollary 18. ■

3.3 Special Cases

It is worth noting some special cases: When $k = l$ and $V^* \neq V$, Algorithm VCC is a 2-approximation algorithm, and the integrality gap for VCC$(k, k, V^*)_{LP}$ has an upper bound of 2. Recall that, as shown in Proposition 10, VCC(k, l, V) is precisely the problem multi-kEC, and we can, without loss of generality, set $l = k$. In this case, we can modify step (4a) of Algorithm VCC by omitting all the edges in $\tilde{J}$ and thus forming $H := (V, E'' \cup J')$. Since this is precisely our Algorithm A from [3], we have an approximation guarantee for VCC(k, l, V) (and an upper bound on the integrality gap for VCC$(k, l, V)_{LP}$) of $\frac{3}{2}$ when k is even, and $\frac{3}{2} + \frac{1}{2k}$ when k is odd.

In the special case $l = \lceil \frac{k}{2} \rceil$, notice that when k is even Algorithm VCC is a $\frac{5}{2}$-approximation algorithm (and we have a $\frac{5}{2}$ upper bound for the integrality gap); and when k is odd the approximation guarantee and upper bound for the integrality gap are at most $\frac{8}{3}$ and asymptotically approach $\frac{5}{2}$ from above as k gets large.

3.3.1 The Special Case VCC($k, 1, V^*$)

As mentioned earlier, the case of $l = 1$, i.e., VCC$(k, 1, V^*)$, is treated separately. In this case, we use the following approximation algorithm:

Algorithm E

Input: A complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, a non-empty subset of vital vertices $V^* \subseteq V$, and non-negative integer $k \in \mathbb{Z}_{\geq 1}$.

1. Using Algorithm M from Section 2.1, find an extended minimum cost spanning tree, M', of G with vital vertices V^*. Let $E' \subset E$ be the edge set of M'. (If $k = 1$, set $H := M'$ and go to Step 4.)

2. Let $\hat{V}^{odd} \subset V$ be the set of vertices of G having odd degree in M'. Find a minimum cost $\hat{T}$-join $\hat{J} \subseteq E$, on G, where $\hat{T} = \hat{V}^{odd}$.

3. Find a minimum cost spanning tree, M, of G. Let $V^{odd} \subset V$ be the set of vertices of G having odd degree in M. Find a minimum cost T-join $J \subseteq E$, on G, where $T = V^{odd}$. Take $\lceil \frac{k-2}{2} \rceil$ copies of the edges in M, and let this form the edge set E'_M. Take $\lfloor \frac{k-2}{2} \rfloor$ copies of the edges in J, and let this form the edge set J'. Take 1 copy of the edges in $\hat{J}$. Form the multi-subgraph $H := (V, E' \cup \hat{J} \cup E'_M \cup J')$.

4. Return H.

Proposition 20 *Algorithm VCC runs in polynomial time and produces a feasible solution of VCC($k, 1, V^*$).*

Proof: A minimum cost spanning tree can be found in $O(|V| \log |V| + |E|)$ time [11]. This becomes $O(|V|^2)$ for a complete graph. Finding a minimum cost T-join can be done in the worst case running time $O(|V|^3)$ [3]. Therefore, using Lemma 6, Algorithm E has worst case running time $O(|V|^3 + |V|^3 + |V|^2 + |V|^2) = O(|V|^3)$.

We now show that the graph H returned by Algorithm E is a feasible solution to VCC($k, 1, V^*$). Clearly, $H[V^*]$ is connected, since it contains the spanning tree $M'[V^*]$. Next, the edges of H partition into 1 copy of the edges of M', 1 copy of the edges in $\hat{J}$, $\lceil \frac{k-2}{2} \rceil$ copies of the edges of M, and $\lfloor \frac{k-2}{2} \rfloor$ copies of the edges in J. Thus, the edges of H partition into a 2-edge connected subgraph of G and a $(k-2)$-edge connected subgraph of G. Hence, H is $(2+k-2) = k$-edge connected. The result follows. ■

The following lemma gives an improvement on the bound of Lemma 14 in the case where $l = 1$:

Lemma 21 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, a non-empty subset of vital vertices $V^* \subseteq V$, and a non-negative integer $k \in \mathbb{Z}$, $k \geq 1$, the following holds on G:*

$$opt(\text{EMST}(V^*)) \leq \; opt(\text{VCC}(k, 1, V^*)_{LP}) \leq \; opt(\text{VCC}(k, 1, V^*)).$$

Proof: Since EMST(V^*) is precisely the special case VCC($1, 1, V^*$), we have that $opt(EMST(V^*)_{LP}) \leq \; opt(VCC(k, 1, V^*)_{LP})$. However, by Corollary 9, the cost of an optimal solution to EMST(V^*) is equal to the optimal value of its linear programming relaxation. The result follows. ■

Theorem 22 (Upper Bound of Integrality Gap of VCC($k, 1, V^*$)$_{LP}$) *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, let $V^* \subseteq V$ be a non-empty set of vital vertices, and let $k \in \mathbb{Z}_{\geq 1}$ be a non-negative integer. The integrality gap for the linear programming relaxation of VCC($k, 1, V^*$) is at most $\frac{5}{2} - \frac{2}{k}$ when k is even, and at most $\frac{5}{2} - \frac{3}{2k}$ when k is odd.*

Proof: Let H be the feasible solution of VCC$(k, 1, V^*)$ that is returned by Algorithm E on G. Using Lemma 12, Lemma 21, and Corollary 15 we have,

$$\begin{aligned}
cost(H) &= 1 \cdot opt_G(EMST(V^*)) + 1 \cdot opt_G(\hat{T}\text{-}join) \\
&\qquad + \left\lfloor \frac{k-2}{2} \right\rfloor \cdot opt_G(T\text{-}join) + \left\lceil \frac{k-2}{2} \right\rceil \cdot opt_G(MST) \\
&\leq \left(1 \cdot 1 + \left(1 + \left\lfloor \frac{k-2}{2} \right\rfloor\right) \cdot \frac{1}{k} + \left\lceil \frac{k-2}{2} \right\rceil \cdot \frac{2}{k}\right) \\
&\qquad \cdot opt_G(VCC(k, 1, V^*)_{LP}). \qquad (27)
\end{aligned}$$

When k is even, (27) becomes

$$\begin{aligned}
cost(H) &\leq \left(1 + \left(1 + \frac{k-2}{2}\right) \cdot \frac{1}{k} + \left(\frac{k-2}{2}\right) \cdot \frac{2}{k}\right) \cdot opt_G(VCC(k, 1, V^*)_{LP}) \\
&= \left(\frac{5}{2} - \frac{2}{k}\right) \cdot opt_G(VCC(k, 1, V^*)_{LP}). \qquad (28)
\end{aligned}$$

When k is odd, (27) becomes

$$\begin{aligned}
cost(H) &\leq \left(1 + \left(1 + \frac{k-2-1}{2}\right) \cdot \frac{1}{k} + \left(\frac{k-2+1}{2}\right) \cdot \frac{2}{k}\right) \\
&\qquad \cdot opt_G(VCC(k, 1, V^*)_{LP}) \\
&= \left(\frac{5}{2} - \frac{3}{2k}\right) \cdot opt_G(VCC(k, 1, V^*)_{LP}). \qquad (29)
\end{aligned}$$

Since $opt_G(VCC(k, l, V^*)) \leq cost(H)$, the result follows. ■

The next corollary follows directly from Theorem 22 and its proof.

Corollary 23 *Let H be the feasible solution of VCC$(k, 1, V^*)$ that is returned by Algorithm E on G. Let $\alpha = \frac{5}{2} - \frac{2}{k}$ when k is even, and let $\alpha = \frac{5}{2} - \frac{3}{2k}$ when k is odd. On G,*

$$cost(H) \leq \alpha \cdot opt(\mathrm{VCC}(k, 1, V^*)_{LP}) \leq \alpha \cdot opt(\mathrm{VCC}(k, 1, V^*)).$$

Proof: The first inequality follows directly from equations (28), and (29). Noting that $opt(VCC(k, l, V^*)_{LP}) \leq opt(VCC(k, l, V^*))$ completes the proof. ■

Corollary 24 (Approximation Guarantee of Algorithm E) *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, let $V^* \subseteq V$ be a non-empty set of vital vertices, and let $k \in \mathbb{Z}_{\geq 1}$ be a non-negative integer. Algorithm E is a $(\frac{5}{2} - \frac{2}{k})$-approximation algorithm for VCC$(k, 1, V^*)$ when k is even and a $(\frac{5}{2} - \frac{3}{2k})$-approximation algorithm when k is odd. In particular, Algorithm E is an $\frac{5}{2}$-approximation algorithm for VCC$(k, 1, V^*)$.*

Proof: Follows directly from Proposition 20 and Corollary 23. ■

From Theorem 22 and Corollary 24, notice that the special case VCC$(2, 1, V^*)$ (in which the secondary vertices are not a cut set) has a $\frac{3}{2}$ approximation guarantee and a $\frac{3}{2}$ upper bound on the integrality gap. This generalizes our previous $\frac{3}{2}$ results for multi-2EC [3]. In particular, when there is only one secondary vertex $r \in V$, VCC$(2, 1, V \backslash \{r\})$ has a $\frac{3}{2}$ approximation guarantee and finds a feasible multi-2EC solution in which r is not a cut vertex.

Lastly, notice that when $l = 1$ and $k = 1$, we get an approximation guarantee and upper bound of 1. This is as expected, since this is just the problem EMST.

VCC Heuristic from Combining Approximation Algorithms

We briefly examine approximation algorithms for VCC that can be obtained by combining known heuristics for other problems, and note that our approximation algorithms VCC and E do better in all instances. First, we need a few upper bounds. By Lemmas 6 and 14,

$$opt_{G[V^*]}(MST) \;\leq\; opt_G(EMST(V^*)) \;\leq\; \frac{2}{l}\, opt_G(VCC(k, l, V^*)). \qquad (30)$$

Since the cost of an optimal multi-lEC solution on $G[V^*]$ is not greater than the cost of an optimal solution of VCC(k, l, V^*) on G restricted to $G[V^*]$,

$$opt_{G[V^*]}(\text{multi-}l\text{EC}) \;\leq\; opt_G(VCC(k, l, V^*)). \qquad (31)$$

Since multi-kEC is a relaxation of VCC(k, l, V^*),

$$opt_G(\text{multi-}k\text{EC}) \;\leq\; opt_G(VCC(k, l, V^*)). \qquad (32)$$

Lastly, the approximation guarantee for multi-kEC using our Algorithm A from [3] is $\frac{3}{2}$, for k even; and $\frac{3}{2} + \frac{1}{2k}$, for k odd ([3], Theorem 5).

When $l = 1$, heuristic solutions for VCC$(k, 1, V^*)$, can be obtained by (a): finding an optimal MST solution on $G[V^*]$, using our Algorithm A from [3] to find a feasible multi-kEC solution on G, and combining them together; or by (b): finding an optimal EMST(V^*) solution on G, using our Algorithm A from [3] to find a feasible multi-$(k-1)$EC solution on G, and combining them together. Using the inequality (30) and the approximation guarantee from [3] for multi-kEC, when k is even method (a) yields a feasible VCC(k, l, V^*) solution with cost less than or equal to $\frac{7}{2}$ of the cost of an optimal VCC(k, l, V^*) solution; and when k is odd method (b) yields a feasible VCC(k, l, V^*) solution with cost less than or equal to $\frac{7}{2}$ of the cost of an optimal VCC(k, l, V^*) solution. Algorithm E gives a better approximation guarantee than this for all values of k.

When $l \geq 2$, a heuristic solution for VCC$(k, 1, V^*)$ can be obtained by (c): using our Algorithm A from [3] to find a feasible multi-lEC solution on $G[V^*]$, using our Algorithm A from [3] to find a feasible multi-kEC solution on G, and combining them together. Using the above mentioned approximation guarantee for Algorithm A for multi-kEC ([3], Theorem 5), and the inequalities (31) and (32), method (c) is a $\frac{3}{2}+\frac{3}{2}=3$-approximation algorithm for VCC(k,l,V^*) when k and l are even; a $(\frac{3}{2}+\frac{1}{2l})+\frac{3}{2}=(3+\frac{1}{2l})$-approximation algorithm for VCC(k,l,V^*) when k is even and l is odd; a $\frac{3}{2}+(\frac{3}{2}+\frac{1}{2k})=(3+\frac{1}{2k})$-approximation algorithm for VCC(k,l,V^*) when l is even and k is odd; and a $(\frac{3}{2}+\frac{1}{2l})+(\frac{3}{2}+\frac{1}{2k})=(3+\frac{1}{2l}+\frac{1}{2k})$-approximation algorithm for VCC(k,l,V^*) when k and l are odd. Comparing these with Theorem 19, Algorithm VCC gives a strictly better approximation guarantee for all instances. Appendix A presents the approximation guarantees of Algorithm VCC for certain instances of VCC, and lists them against those of method (c). Not only this, but also notice that method (c) does not yield an upper bound on the integrality gap for the LP relaxation of VCC.

Notice that finding a feasible multi-lEC solution on $G[V^*]$, finding a feasible multi-kEC solution on the graph with V^* identified as a single vertex (i.e., on G/V^*), and then combining them together does not necessarily solve VCC(k,l,V^*).

4 Extension: Multiple Vital Cores

Instead of being defined for just one vital core, EMST can be generalized to include numerous disjoint vital cores as follows: Given a complete graph, $G=(V,E)$, with non-negative edge costs $c\in\mathbb{R}^E$, and a set of given non-empty pairwise disjoint subsets of vital vertices $\mathcal{V}:=\{V_1^*,V_2^*,\ldots,V_q^*\}$ such that $V_m^*\subseteq V$ for $m=1,2,\ldots,q$, find a minimum cost spanning tree of G that remains connected on each component $G[V_1^*],G[V_2^*],\ldots,G[V_q^*]$ when the secondary vertices $V':=V\backslash(V_1^*\cup V_2^*\cup\cdots\cup V_q^*)$ are removed; i.e., whose induced subgraphs on each of the vital cores $G[V_1^*],G[V_2^*],\ldots,G[V_q^*]$ are themselves a spanning tree on that vital core. We call this generalization of EMST the *vital core minimum spanning tree problem* (VCMST). Instances of VCMST are denoted by VCMST$(\mathcal{V})$. Clearly, if G is not connected, there is no solution to VCMST. Also, if G is connected but not complete, we can make it complete by adding in the "missing" edges and assigning them each a very large edge cost. Clearly, without loss of generality, we can assume G is simple. When $\mathcal{V}$ consists of a single subset $V^*\subseteq V$, VCMST reduces to EMST. Thus, we have the following inequality:

Proposition 25 *Given a complete graph $G=(V,E)$ with non-negative edge costs $c\in\mathbb{R}^E$, and a set of non-empty disjoint subsets of vertices $V_1^*,V_2^*,\ldots,V_q^*\subseteq V$, the following holds on G: opt(EMST) $\leq$ opt(VCMST).*

In this section, we present a polynomial-time algorithm for solving VCMST, as well as a complete linear description for VCMST. This generalizes EMST results from Section 2.

The following characteristic of a feasible solution of VCMST is used by the algorithm for VCMST($\mathcal{V}$):

Lemma 26 *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, and let $\mathcal{V} = \{V_1^*, V_2^*, \dots, V_q^*\}$ be a set of non-empty pairwise disjoint subsets of vertices such that $V_m^* \subseteq V$ for $m = 1, 2, \dots, q$. Let M be an optimal solution to VCMST($\mathcal{V}$) on G. Then $M[V_m^*]$ is a minimum cost spanning tree of $G[V_m^*]$, for $m = 1, 2, \dots, q$.*

Proof: Consider the subset of vital vertices V_m^*, $m \in \{1, 2, \dots, q\}$. Since the removal of the secondary vertices V' does not disconnect $M[V_m^*]$, therefore, $M[V_m^*]$ is a connected subgraph of $G[V_m^*]$. Moreover, since M is a spanning tree of G, therefore $M[V_m^*]$ is a spanning subgraph of $G[V_m^*]$ and is acyclic. Since $M[V_m^*]$ is a connected, acyclic, spanning subgraph of the vital core $G[V_m^*]$; thus, $M[V_m^*]$ is a spanning tree of $G[V_m^*]$. Since this holds for all $m \in \{1, 2, \dots, q\}$, therefore $M[V']$ has connected components $T_1, \dots, T_t$ (for some $t \in \mathbf{N}_{>0}$) which are each trees, and which are each connected in M to each $M[V_m^*]$ by at most one distinct edge in $\delta_M(M[V_m^*])$, respectively.

Suppose B_m is a minimum cost spanning tree of $G[V_m^*]$. Thus, $cost(B_m) \leq cost(M[V_m^*])$. Suppose $cost(B_m) < cost(M[V_m^*])$. Let $\hat{M}$ be the subgraph of G obtained from M by replacing $M[V_m^*]$ by B_m. See Figure 4. It is easy to see that $\hat{M}$ is a spanning tree of G. Furthermore, $\hat{M}[V_m^*]$ is connected, since $\hat{M}[V_m^*]$

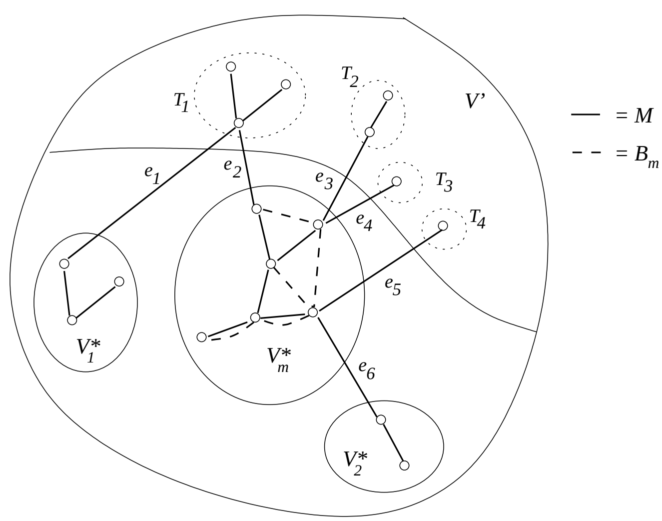


Figure 4: Replacing $M[V^*]$ by B_m in M.

consists precisely of B_m (which is connected). Thus, $\hat{M}$ is a feasible solution to VCMST($\mathcal{V}$) on G, and $cost(\hat{M}) < cost(M)$, which is a contradiction with the fact that M is an optimal solution to VCNST($\mathcal{V}$) on G. Thus, $cost(B_m) \not<$

$cost(M[V_m^*])$. Therefore, $M[V_m^*]$ is a minimum cost spanning tree of $G[V_m^*]$, for $m = 1, 2, \ldots, q$. ∎

Notice, from Lemma 26, that in a feasible solution of VCMST, every secondary vertex $v \in V'$ is either a leaf of the spanning tree, or is part of a branch of the spanning tree such that from v to the leaf/leaves of the branch all the vertices are secondary vertices.

Given a graph $G = (V, E)$, and $\mathcal{V} = \{V_1^*, V_2^*, \ldots, V_q^*\}$ for non-empty disjoint subsets $V_m^* \subseteq V$, $m = 1, 2, \ldots, q$, define the multi-graph $G/\mathcal{V}$ to be the shrunk graph of G with respect to $\mathcal{V}$, obtained by identifying all the vertices in V_m^* into a single ("shrunk") vertex w_m and deleting all the edges of G that have both endpoints in V_m^*; for $m = 1, 2, \ldots, q$. Notice that $G/\mathcal{V}$ has $|V| - \sum_{i=1}^{q} |V_i^*| + q$ vertices, and its edge set, $E_{\mathcal{V}}$, consists of all the edges of G that have exactly one or no endpoint in $\mathcal{V}$. Notice that edges iw_m, $i \in V'$, in $G/\mathcal{V}$ are in 1 to 1 correspondence with edges $ij \in E$, $j \in V_m^*$, $m \in \{1, 2, \ldots, q\}$. Edges $w_p w_m$ in $G/\mathcal{V}$, $m, p \in \{1, 2, \ldots, q\}$, $m \neq p$, are in 1 to 1 correspondence with the edges in G that have one endpoint in V_m^* and one endpoint in V_p^*. Edges in $G/\mathcal{V}$ that are not incident to any of the shrunk vertices w_m are in 1 to 1 correspondence with the edges in G that do not have any endpoints in V_m^*, for all $m = 1, 2, \ldots, q$.

Lemma 27 *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, and let $\mathcal{V} = \{V_1^*, V_2^*, \ldots, V_q^*\}$ be a set of non-empty disjoint subsets of vertices such that $V_m^* \subseteq V$ for $m = 1, 2, \ldots, q$. Let $V' = V \backslash \mathcal{V}$. Let M be an optimal solution to VCMST($\mathcal{V}$) on G. Then M restricted to the shrunk graph $G/\mathcal{V}$ is a minimum cost spanning tree of $G/\mathcal{V}$.*

Proof: Notice that $G/\mathcal{V}$ is connected (since G is connected), and thus contains a spanning tree. Let M^* be M restricted to the shrunk graph $G/\mathcal{V}$. Since, by Lemma 26, each $M[V_m^*]$ is a spanning tree of $G[V_m^*]$, for $m = 1, 2, \ldots, q$; it is easy to see that M^* is a spanning tree of $G/\mathcal{V}$. See Figure 4.

Suppose A is a minimum cost spanning tree of $G/\mathcal{V}$. Thus, $cost(A) \leq cost(M^*)$. Suppose $cost(A) < cost(M^*)$. Let $\hat{M}$ be the subgraph of G obtained from M by replacing the edges of M^* by the edges of A. Clearly, $\hat{M}$ is a spanning tree of G. By construction, $cost(\hat{M}) = \sum_{i=1}^{q} cost(M[V_i^*]) + cost(A) < \sum_{i=1}^{q} cost(M[V_i^*]) + cost(M^*) = cost(M)$. This is a contradiction with M a minimum cost spanning tree of G. Thus, $cost(A) = cost(M^*)$, and M^* is a minimum cost spanning tree of $G/\mathcal{V}$. ∎

Notice that, from Lemmas 26 and 27, any multiple copies of edges in G can be removed, keeping only, for each pair of distinct vertices, the edge joining them which has the cheapest edge cost. Thus, in considering VCMST, without loss of generality we can always assume that G is simple.

Based on Lemmas 26 and 27 above, we have the following polynomial-time algorithm for VCMST($\mathcal{V}$):

Algorithm GM

Input: A complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, a set of vital cores $\mathcal{V} = \{V_1^*, V_2^*, \ldots, V_q^*\}$, such that $V_m^* \cap V_p^* = \emptyset$ for distinct $m, p \in \{1, 2, \ldots, q\}$, and $V_m^* \subseteq V$ for $m = 1, 2, \ldots, q$; and the set of secondary vertices $V' := V \backslash \mathcal{V}$.

G1. For $m = 1, 2, \ldots, q$, find a minimum cost spanning tree, $\tilde{M}_m$, of the vital core $G[V_m^*]$.

G2. Find a minimum cost spanning tree, M^*, of the shrunk graph $G/\mathcal{V}$.

G3. Form the subgraph $M' = \big(V, E(\tilde{M}_1) \cup \cdots \cup E(\tilde{M}_q) \cup E(M^*)\big)$.

Lemma 28 *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, and let $\mathcal{V} = \{V_1^*, V_2^*, \ldots, V_q^*\}$ be a set of non-empty pairwise disjoint subsets of vertices such that $V_m^* \subseteq V$ for $m = 1, 2, \ldots, q$. Let $V' = V \backslash \mathcal{V}$. The subgraph M' returned by Algorithm GM is an optimal solution to VCMST($\mathcal{V}$) on G, and can be constructed in worst case running time $O(|V|^2)$.*

Proof: By Lemmas 26 and 27, an optimal solution to VCMST($\mathcal{V}$) on G consists exactly of the vertices in V along with the edges of a minimum cost spanning tree of each of $G[V_1^*], \ldots G[V_q^*]$ and the edges of a minimum cost spanning tree of $G/\mathcal{V}$.[5] This is precisely M'. Thus, M' is an optimal solution to VCMST($\mathcal{V}$) on G.

For a connected graph with n vertices and m edges, a minimum cost spanning tree can be found in $O(n \log n + m)$ time [11]. This becomes $O(n^2)$ for a complete graph. Thus, steps (G1) and (G2) can be implemented in worst case running time $O(|V|^2)$. ■

A complete linear description of EMST is given in Section 2.2. Using the same techniques as above, this can be extended naturally to form a complete

[5] Notice that, without loss of generality, $G/\mathcal{V}$ can be made into a simple graph by removing, for each secondary vertex $u \in V'$, all the multiple copies of edges uv, $v \in \{w_1, \ldots, w_q\} \cup V'$, except for the copy of uv with the cheapest edge cost. Thus, any algorithm for finding a minimum cost spanning tree can be used, regardless of whether or not that algorithm starts on a simple or multi-graph.

linear description of VCMST:

$$\begin{aligned}
\text{minimize} \quad & cx && (33)\\
\text{subject to} \quad & \sum_{e \in E_P} x_e \;\geq\; k_P - 1, \text{ for all partitions } P \text{ of } G/\mathcal{V}, && (34)\\
& \sum_{e \in E_P} x_e \;\geq\; k_P - 1, \text{ for all partitions } P \text{ of } && (35)\\
& \qquad\qquad G[V_1^*], G[V_2^*], \ldots, G[V_q^*], \text{ respectively,} &&\\
& x_e \;\geq\; 0, \quad \text{for all } e \in E. && (36)
\end{aligned}$$

Constraints (34) and (36) are precisely those constraints for the minimum spanning tree LP(1) on the shrunk graph $G/\mathcal{V}$ of G, and constraints (35) and (36) are precisely those constraints for the minimum spanning tree LP(1) on the vital cores $G[V_1^*], G[V_2^*], \ldots, G[V_q^*]$, respectively.

5 Conclusion

We have shown that in the case that $k = l = 1$, VCC results in a useful generalization of the minimum cost spanning tree problem; a generalization for which we provide a complete linear description as well as a polynomial-time algorithm. We have also presented an approximation algorithm for VCC(k, l, V^*), and have presented an upper bound for the integrality gap for its linear programming relaxation. These results for VCC are the first such to be presented. Although a better guarantee than that of Algorithm VCC can be obtained for large differences between k and l, Algorithm VCC yields the best guarantee both for smaller differences between k and l and for small connectivities.

For VCC(k, l, V^*) with $k > l > \lceil \frac{k}{2} \rceil$, as an upper bound Algorithm VCC has an approximation guarantee (and upper bound on the integrality gap) of $\frac{8}{3}$ for $l \geq \lceil \frac{k}{2} \rceil$, $\frac{19}{6}$ for $2 \leq l < \lceil \frac{k}{2} \rceil$, and $\frac{5}{2}$ for $l = 1$ or for $l = \lceil \frac{k}{2} \rceil$ and k even. When $l = \lceil \frac{k}{2} \rceil$ and k is odd the approximation guarantee and upper bound for the integrality gap are at most $\frac{8}{3}$ and asymptotically approach $\frac{5}{2}$ from above as k gets large. For the special case VCC(k, k, V^*), $V^* \neq V$, our approximation guarantee is 2. When $l < \lceil \frac{k}{2} \rceil$, Algorithm VCC is very similar to our previous approximation algorithm (Algorithm A) for, in particular, multi-kEC [3]. This is because when $l < \lceil \frac{k}{2} \rceil$, for an optimal solution H of VCC(k, l, V^*), $H[V^*]$ is $\lceil \frac{k}{2} \rceil$-edge connected, and thus is certainly l-edge connected. The only difference in this case between Algorithm A for multi-kEC [3] and Algorithm VCC (for VCC(k, l, V^*)) is that Algorithm A uses copies of an MST of G, whereas Algorithm VCC uses copies of an EMST(V^*) of G. Thus, from Proposition 3, the approximation guarantee of Algorithm VCC is expected to be at least that of Algorithm A, which is indeed the case. In the special case VCC(k, l, V), which is simply the problem multi-

kEC, we have an approximation algorithm with approximation guarantee of $\frac{3}{2}$ when k is even, and $\frac{3}{2} + \frac{1}{2k}$ when k is odd (agreeing with our previous results [3]). Thus, Algorithm VCC generalizes our previous approximation algorithm for multi-kEC.

Additionally, it is interesting to note that when $l = k - 1$, Algorithm VCC has an approximation guarantee of $\frac{3}{2} + \frac{1}{2}(\frac{k}{k-1})$ for k even, and $\frac{3}{2} + \frac{1}{2}(\frac{k^2+1}{k^2-1})$ for k odd. In particular, when there is only one secondary vertex $r \in V$, Algorithm VCC is a $\frac{3}{2}$-approximation algorithm for VCC$(2, 1, V \backslash \{r\})$, i.e., in which r is not a cut vertex. For the special case VCC$(2, 1, V^*)$ (in which the secondary vertices are not a cut set), Algorithm E has a $\frac{3}{2}$ approximation guarantee, generalizing our previous $\frac{3}{2}$-approximation algorithm for multi-2EC [3].

Finally, observe that our approximation algorithm for VCC has approximation guarantee (and upper bound on the integrality gap) that is strictly less than 2 for the special cases VCC$(2, 1, V^*)$ and VCC(k, l, V).

Lastly, we have provided a complete linear description of, and a polynomial-time algorithm for, VCMST (the extension of EMST to numerous disjoint vital cores).

References

[1] P. Angeloudis, D. Fisk, Large Subway Systems as Complex Networks, Physica A 367 (2006) 553-558.

[2] V. Auletta, Y. Dinitz, Z. Nutov, A 2-Approximation Algorithm for Finding an Optimum 3-Vertex-Connected Spanning Subgraph, J. of Algorithms 32 (1999) 21-30.

[3] S. Boyd, A. Cameron, Integrality Gap and Approximation Algorithm for the Multi-survivable Network Design Problem, University of Ottawa, Technical Report TR-2009-03 (2009).

[4] S. Chopra, On the Spanning Tree Polyhedron, Operations Research Letters 8 (1989) 25-29.

[5] W.J. Cook, W.H. Cunningham, W.R. Pulleyblank, A. Shrijver, Combinatorial Optimization, Wiley Interscience (1998).

[6] K.P. Eswaran, R.E. Tarjan, Augmentation Problems, SIAM J. Comput. 5 (1976) 653-665.

[7] G. Even, J. Feldman, G. Kortsarz, Z. Nutov, A 3/2-Approximation Algorithm for Augmenting the Edge-Connectivity of a Graph from 1 to 2 Using a Subset of a Given Edge Set, APPROX-RANDOM 2001, LNCS2129 (2001) 90101.

[8] G. Even, G. Kortsarz, Z. Nutov, A 1.5-approximation for Augmenting a Connected Graph into a Two-connected Graph, Inf. Process. Lett. 111(6) (2011) 296-300.

[9] L. Fleischer, K. Jain, D.P. Williamson, Iterative Rounding 2-approximation Algorithms for Minimum-cost Vertex Connectivity Problems, Journal of Computer and System Sciences 72 (2006) 838867.

[10] G.N. Frederickson, J. Ja'Ja', On the Relationship Between the Biconnectivity Augmentation and Travelling Salesman Problems, Theoretical Computer Science 19 (1982) 189-201.

[11] M.L. Fredman, R.E. Tarjan, Fibonacci Heaps and Their Uses in Improved Network Optimization Algorithms, Journal of the Association for Computing Machinery 34 No.3 (1987) 596-615.

[12] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman & Co., New York, NY (1979).

[13] M.X. Goemans and D.J. Bertsimas, Survivable Networks, Linear Programming Relaxations and the Parsimonious Property, Mathematical Programming 60 (1993) 145-166.

[14] K. Jain, A Factor 2 Approximation Algorithm for the Generalized Steiner Network Problem, Combinatorica 21 No.1 (2001) 39-60.

[15] K. Jain, I. Mandoiu, V. V. Vazirani, D.P. Williamson: A Primal-dual Schema Based Approximation Algorithm for the Element Connectivity Problem, Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms (1999) 484489.

[16] G. Kortsarz, Z. Nutov, Approximating Node Connectivity Problems via Set Covers, Algorithmica 37 (2003) 7592.

[17] G. Kortsarz, Z. Nutov, Tight Approximation Algorithm for Connectivity Augmentation Problems, Journal of Computer and System Sciences 74 (2008) 662670.

[18] G. Leberman, Connectivity Augmentation Problems, M.Sc. Thesis, The Open University of Israel (2005).

[19] T. Mashima, S. Taoka, T. Waranabe, A 2-Approximation Algorithm to $(k+1)$-Edge-Connect a Specified Set of Vertices in a k-Edge-Connected Graph, IEICE Trans. Fundamentals, V.E88A N.5 (2005) 1290-1300.

[20] Z. Nutov, Approximating Connectivity Augmentation Problems, SODA '05 Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms (2005) 176-185.

[21] J. Ramirez-Marquez, C. Rocco, G. Levitin, Optimal Protection of General Sourcesink Networks Via Evolutionary Techniques, Reliability Engineering & System Safety 94 Issue 10 (2009), 1676-1684.

Appendix A: Comparison of Approximation Guarantees

Near the end of Section 3.3, we briefly examined some approximation algorithms for VCC that can be obtained by combining known heuristics for other problems, and we noted that our approximation Algorithms VCC and E do better in all instances. Among combinations of known heuristics that we mentioned, we noted what we called Method (c): When $l \geq 2$, a heuristic solution for VCC$(k, 1, V^*)$ can be obtained by using our Algorithm A from [3] to find a feasible multi-lEC solution on $G[V^*]$, using our Algorithm A from [3] to find a feasible multi-kEC solution on G, and combining them together. Method (c) is a 3-approximation algorithm for VCC(k, l, V^*) when k and l are even; a $(3 + \frac{1}{2l})$-approximation algorithm for VCC(k, l, V^*) when k is even and l is odd; a $(3+\frac{1}{2k})$-approximation algorithm for VCC(k, l, V^*) when l is even and k is odd; and a $(3 + \frac{1}{2l} + \frac{1}{2k})$-approximation algorithm for VCC(k, l, V^*) when k and l are odd. Algorithm VCC gives a strictly better approximation guarantee for all instances.

The below table presents the approximation guarantees of Algorithm VCC for certain instances of VCC, and lists them against those of Method (c). In the special case that $l = 1$, Algorithm E has an approximation guarantee of 1 when $k = 1$; of $\frac{3}{2}$ when $k = 2$; of 2 when $k = 3, 4$; of $\frac{11}{5}$ when $k = 5$; of $\frac{13}{6}$ when $k = 6$; etc.

	l	k	Approx. Guarantee of Algorithm VCC	Approx. Guarantee of method (c)
$k = l$			**2**	3
$k = 2l$			**5/2**	3
k, l even	2	6	**8/3**	3
		...		
	4	6	**9/4**	3
		10	**13/5**	3
		...		
	6	8	**13/6**	3
		10	**14/6**	3
		14	**18/7**	3
		16	**21/8**	3
		...		
	...	...		
k even, l odd	3	4	**13/6**	19/6
		8	**67/24**	19/6
		...		
	5	6	**21/10**	31/10
		8	**23/10**	31/10
		12	**161/60**	31/10
		14	**96/35**	31/10
		...		
	7	8	**29/14**	43/14
		10	**31/14**	43/14
		12	**33/14**	43/14
		16	**295/112**	43/14
		18	**169/63**	43/14
		...		
	...	...		

	l	k	Approx. Guarantee of Algorithm VCC	Approx. Guarantee of method (c)
k odd, l even	2	3	**7/3**	19/6
		5	**27/10**	31/10
		...		
	4	5	**43/20**	31/10
		7	**17/7**	43/14
		9	**47/18**	55/18
		...		
	6	7	**44/21**	43/14
		9	**41/18**	55/18
		11	**27/11**	67/22
		13	**67/26**	79/26
		...		
	...	...		
k, l odd	3	5	**12/5**	49/15
		7	**8/3**	68/21
		9	**25/9**	29/9
		...		
	5	7	**78/35**	111/35
		9	**22/9**	142/45
		11	**13/5**	173/55
		13	**174/65**	204/65
		...		
	7	9	**136/63**	197/63
		11	**178/77**	240/77
		13	**32/13**	283/91
		15	**18/7**	326/105
		17	**313/119**	369/119
		...		