Approximation Algorithms for Network Connectivity Problems

Amy Cameron

Thesis submitted to the Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the degree of
Doctorate in Mathematics[1].

Department of Mathematics and Statistics
Faculty of Science
University of Ottawa

[1]The Doctorate in Mathematics program is a joint program with Carleton University, administered by the Ottawa-Carleton Institute of Mathematics and Statistics.

Abstract

In this dissertation, we examine specific network connectivity problems, and achieve improved approximation algorithm and integrality gap results for them. We introduce an important new, highly useful and applicable, network connectivity problem – the Vital Core Connectivity Problem (VCC). Despite its many practical uses, this problem has not been previously studied. We present the first constant factor approximation algorithm for VCC, and provide an upper bound on the integrality gap of its linear programming relaxation. We also introduce a new, useful, extension of the minimum spanning tree problem, called the Extended Minimum Spanning Tree Problem (EMST), that is based on a special case of VCC; and provide both a polynomial-time algorithm and a complete linear description for it. Furthermore, we show how to generalize this new problem to handle numerous disjoint vital cores, providing the first complete linear description of, and polynomial-time algorithm for, the generalized problem.

We examine the Survivable Network Design Problem (SNDP) with multiple copies of edges allowed in the solution (multi-SNDP), and present a new approximation algorithm for which the approximation guarantee is better than that of the current best known for certain cases of multi-SNDP. With our method, we also obtain improved bounds on the integrality gap of the linear programming relaxation of the problem. Furthermore, we show the application of these results to variations of SNDP. We investigate cases where the optimal values of multi-SNDP and SNDP are equal; and we present an improvement on the previously best known integrality gap

bound and approximation guarantee for the special case of SNDP with metric costs and low vertex connectivity requirements, as well as for the similar special case of the Vertex Connected Survivable Network Design Problem (VC-SNDP).

The quality of the results that one can obtain for a given network design problem often depends on its integer linear programming formulation, and, in particular, on its linear programming relaxation. In this connection, we investigate formulations for the Steiner Tree Problem (ST). We propose two new formulations for ST, and investigate their strength in terms of their associated integrality gaps.

To my parents and grandparents,

who have helped “train me in the way I should go”

(Proverbs 22:6).

Acknowledgments

This thesis, like anything else, did not occur in a vacuum. Many people have been involved in it in all kinds of ways.

A huge thank you to my thesis adviser, Dr. Sylvia Boyd. In a graduate course, she first introduced me to the fascinating and very relevant field of combinatorial optimization, imparting her proficiency and enthusiasm for it. Since that time, she has graciously and tirelessly been a wonderful aid, using her expertise in helping and guiding me through to the completion of this dissertation.

Thank you to all my teachers and professors over the years, who have challenged me and taught me many things – both mathematical and otherwise! Warm thanks is extended to my "office-mates", in particular to Dr. Pedro Villanueva, Dr. Maryam Haghighi, Dr. Paul Elliott-Magwood, Mr. Patrick Niesink, Mr. Sebastian Raaphorst and others, who have shared my office and have brought encouragement and laughter throughout this work.

A bit of a mathematical streak seems to run in some of my family. Grandpa Henry (Hume), who taught in a one-room school before joining the RCAF and repairing aircraft radios in WWII, devised a way on his own of how to take a manual cubed root. Grandad Jack (Cameron), who worked in railroad machine shops, devised improvements such as determining the combination of metals and bell shape to give train bells the best-sounding ring. Both were always full of encouragement and great advice, with Grandpa Henry encouraging, "Just keep plugging away at it", and

Grandad Jack reminding me "TTT – Things Take Time". This is the first doctorate in the family, and both sets of my grandparents longed to see this day.

A special thank you to my parents and family. Without you this day might never have occurred. Thank you to all of you for your love, encouragement, and support. My mother was the one who gave me my very first math text – a much-enjoyed and much-used small cardboard flap book which counted from 1 to 10, and which I still remember going through over and over again! My mother was also the one who taught me to regard mathematics as a challenge and puzzle which "simply" requires one to fit the right pieces together. Thank you, Mom and Dad. My brothers and sister are wonderful. As well as being an encouragement, they have loved storing up and reminding me of my "bloopers" that I have said while I was distracted and still puzzling over a problem! Thank you as well to my family and friends, for putting up with me essentially disappearing for long periods of time while I worked (and to my nieces and nephews, who lovingly regard me as their aunt who, like them, is in school, but who seems to have an incomprehensible amount of homework!).

Warm thanks to my church family, and all those who encouraged me and prayed for me. A special thank you to my pastor, Dr. Richard Ganz, for his spiritual care, his teaching, his encouragement and advice, his interest in my work even when he could not understand a word of it, and his introducing me a number of years ago to running, thus keeping me fit and sane. Thanks as well to Dr. Eric Jackson and Dr. John Coombs, who have been wonderful in helping keep me on my feet.

Finally, all of this would have been impossible without my Lord and Saviour, the Lord Jesus Christ, who not only numbers the stars and calls them all by name (Psalm 147:4, Isaiah 40:26), but who also numbers the hairs of our head and calls *us* by name, giving us a new name and a new start when we turn to Him (Luke 12:7, Revelation 2:17, 2 Corinthians 5:17). As the Bible declares: "To the only God our Saviour, through Jesus Christ our Lord, be glory, majesty, dominion and authority, before all time, now and forever. Amen" (Jude 1:24-25).

Contents

List of Figures

Chapter 1

Introduction

1.1 Introduction

From communication lines, to pipelines, to roads, to railways, to electric circuitry, to power lines, networks (and thus the design of networks) play an important role both in our lives and in the functioning of our nation. Disruption of these networks can have numerous and serious consequences, both in terms of cost, and in terms of people's lives. For example, snowstorms, forest fires, avalanches, floods, etc., can occur which block or destroy roads or railway lines or bring down telephone poles. This can isolate whole regions, especially those in remote locations, causing many problems. Failure of a power line can cause loss of power to whole regions or to critical power stations, unless another routing for that power is available. Similarly, failure of a wire in a circuit board can have grave consequences if the wiring is not constructed in a way that can survive such a failure. Fibre optic telecommunication networks rely heavily on having a certain amount of robustness to loss of lines [25, 31]. When this is not the case, substantial loss can occur. For example, in 1993, a large snow storm on the East Coast of the United States brought down the entire nation-wide network of bank machines, causing the banking industry an estimated loss of about one hundred million dollars [57]. Clearly, the network had not been designed with such a disruption in mind, nor had it been designed to limit or contain such a disruption to a given region.

From these few examples, it is easy to see that the construction of survivable networks is of paramount importance. Since we do not wish any of these networks, and many others like them, to break down readily in the event of natural catastrophes (ice storms, earthquakes, etc.) or man-made catastrophes (accidents, terrorism, etc.), it is critical to be able to design reliable networks that can survive certain degrees of loss to either links, centres, or a combination of both. In other words, given a network of centres and specific non-negative costs for connecting any two centres with a link, it is important to find a cheapest way to construct a network so that it can survive the loss of a certain number of links (i.e., so that the network remains connected even if a certain number of the links are lost). In addition, we may also have the requirement that the network has the property that if certain centres are lost the remainder of the network not only remains connected but also can survive the loss of a certain (possibly different) number of links. In designing such networks, we would clearly also prefer the networks to be constructed at the lowest possible cost, since finding a solution that is either optimal or very close to optimal means substantial financial savings for companies constructing such networks.

In this context of network design, "cost" can be interpreted as the construction cost, and "reliable" as the continuation of services in the event of a catastrophe causing the loss of a given number of "links" (i.e. communications cables, pipeline segments, or transportation roads). Such a continuation of services is ensured by the network having been designed in such a way that service can be restored by re-routing through other existing centres and links. 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.

Problems dealing with the design of reliable networks lie in the relatively recent field of combinatorial optimization. This field is heavily used in modelling and is related to the field of operations research, which came into being during World War II. Among other things, combinatorial optimization involves optimization problems (minimization or maximization), linear programming, and approximation algorithms. The applications include network design (transportation, communication, wiring of microchips, neural networks in the brain, etc.), scheduling, and routing, to name a few. For an excellent history of combinatorial optimization up until 1960, see Schrijver's work [56].

In combinatorial optimization, networks are generally modelled using graphs. Graphs were first introduced and used in modelling in the 1700s when the Swiss mathematician Leonhard Euler constructed one for the first time in answering the previously long-standing question of the Königsberg bridges problem[1]. A *graph* $G = (V, E)$ is a structure consisting of a set of vertices V and a set of edges E. In a *simple graph*, every edge joins a unique pair of distinct vertices (so that there are no "loops" where an edge joins a vertex to itself, nor are there any copies of an edge). If there are multiple copies of an edge between at least one pair of distinct vertices in a graph, the graph is called a *multi-graph*. In modelling network design problems, the vertices

[1] Königsberg was the capital city of East Prussia (now Kaliningrad, Russia). The problem was to determine if townsfolk could cross each of the city's seven bridges exactly once and end at the spot they had started (in later mathematical terms, the question was to determine if an Euler tour existed). In solving this question, Euler removed distracting extraneous information by converting the problem to a more abstract representation - what later became known as a graph - in which the bridges (or "links") were drawn as lines, and the islands and distinct pieces of land (or "centres") were drawn as circles. Using a graph representing the problem, Euler proved that the answer to the Königsberg bridges problem was "No" – in order to have a "network" in which all the links can be travelled exactly once and one finishes where one started (i.e., at the same centre, or land mass), all of the centres must be joined to an even number of links.

of a graph represent the centres of the network, and the edges of the graph represent the links of the network. A *complete* graph is one in which there is an edge between every pair of distinct vertices. A *weighted graph* is a graph that has *weights*, or *costs*, on its edges. Such costs represent the cost of building or maintaining that edge in the network. For our purposes, these costs are always non-negative. When there are no costs associated with the edges, the graph is called an *unweighted graph*. A *subgraph* is a graph that consists of a subset of the vertices and edges of a given graph.

As we will see, many network connectivity problems are NP-hard. This implies that they are hard to solve exactly to optimality. It is also considered highly unlikely that a practical and efficient algorithm can ever be found that solves them. For such problems, one aims at obtaining approximate, or near-optimal, solutions. To do this, fast and efficient algorithms are developed which are guaranteed to return a feasible solution that has a total cost within a small constant factor of the cost of the optimal solution. This approach is useful since it can be done without knowing the optimal solution, or even knowing the total cost of the optimal solution. An efficient algorithm that can guarantee a solution whose cost is at most α times the cost of an optimal solution (for some constant $\alpha \geq 1$), is called an *α-approximation algorithm*.

To express a network design problem (represented by a graph) in a way that can be more easily explored mathematically, integer linear programming formulations are often used. The relaxations of such formulations are useful in many solution techniques that are used for NP-hard problems, and provide important lower bounds on the value of the minimum cost solutions. Generally, different integer linear programming (ILP) formulations exist for the same problem, some of which have linear programming (LP) relaxations that yield better results for the problem than do others. Thus, we also examine *integrality gaps* – a way of measuring how "good" the lower bound provided by a certain linear programming relaxation is for a given problem.

This dissertation examines some specific network connectivity problems, with the goal of achieving improved approximation algorithm and integrality gap results for these problems. This dissertation also introduces an important new, highly useful and applicable, network connectivity problem, and presents an approximation algorithm and integrality gap for it (as well as other results).

One of the specific problems we study is the *(edge connected) survivable network design problem* (SNDP). This problem consists of finding a cheapest way to construct a network so that it can survive the loss of certain numbers of links in certain areas of the network. This is used, for example, in designing fibre optic telecommunications networks [2, 25, 32]. More specifically, given a complete graph $G = (V, E)$ with non-negative edge costs, and vertex connectivity requirements r_i on all its vertices $i \in V$, we let the edge connectivity requirement r_{ij} of each edge $ij \in E$ be the minimum vertex requirement of its two endpoints. Problem SNDP is then the problem of finding a minimum cost subgraph that has, for each pair of distinct vertices i and j, at least r_{ij} edge-disjoint paths between those two vertices. In a feasible solution of SNDP, the loss of any $0 < l < \max_{ij \in E} r_{ij}$ edges still allows all vertices with vertex connectivity requirement greater than l to be connected to each other. If SNDP allows multiple copies of the edges to be chosen in the solution, it is denoted by *multi-SNDP*. When the edge costs $c \in \mathbb{R}^E$ of a graph are non-negative and satisfy the triangle inequality $c_{uw} \leq c_{uv} + c_{vw}$ for all distinct vertices $u, v, w \in V$, the costs are called *metric*. Problem SNDP with metric edge costs is denoted by *metric SNDP*. In its general form, SNDP was first studied in the early 1990's [2, 28]. A brief summary of some of the previously known approximation algorithms for SNDP is given in [1]. Currently, the best general constant factor approximation guarantee for multi-SNDP and SNDP is 2 [37].

If, instead of edge-disjoint paths, we require a minimum cost subgraph that has, for each pair of distinct vertices i and j, at least r_{ij} internally vertex-disjoint paths between those two vertices, then the problem is that of the *vertex connected survivable network design problem* (VC-SNDP). Problem VC-SNDP with metric edge costs is denoted by *metric VC-SNDP*. There is a 2-approximation algorithm for VC-SNDP with vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$ [20]; however, no constant factor approximation algorithm exists for the general VC-SNDP [20]. For a short summary of some of the known results on VC-SNDP, see [20, 50].

A special case of SNDP and VC-SNDP is the well-known *Steiner tree problem* (ST). Given a graph and a subset of its vertices (vertices in this subset are called *terminal vertices*), this is the problem of finding a minimum cost connected subgraph that connects the terminal vertices and contains no cycles. It may include all, some, or none of the non-terminal vertices.

Problem SNDP encompasses many network design problems other than ST. In its general form, it ensures that certain parts of the network remain connected even if a certain number of the network's links are lost. However, this does not capture networks that have the additional requirement that if certain centres are lost the remainder of the network not only is connected but also can survive the loss of a certain (possibly different) number of links. The usefulness and importance of such a network can be seen in its applications to transportation systems such as subway systems, as well as to military networks. Large subway systems, as with other complex networks, are generally quite robust to random failure or attack [3, 5]. However, since centres joining many links usually have an important function in the overall connectivity of a network, subway stations with a high number of subway lines joined to them are more vulnerable to targeted attack [5]. In such cases, we would like the entire network to have a given edge connectivity, and also ensure that the loss of certain more vulnerable stations not only does not disconnect the network but also leaves the network with a given guaranteed edge connectivity. This could be less than the overall edge connectivity, since a higher degree of connectivity is not required until the vulnerable stations are restored and the entire network is back in operation. This guaranteed edge connectivity of the remaining network makes it harder for immediate subsequent attacks to disconnect it, and also forces a much larger number of simultaneous attacks on different stations and lines in order to substantially disrupt the network, and thus making such attacks far more difficult to perform. The value of having such a network is demonstrated by the March 2004 Madrid and July 2005 London subway bombings [5, 54].

The importance of such a reliable network extends very clearly and significantly to our military as well. For instance, in combat operations, military outposts are often placed in key locations such as mountain ranges, etc. These outposts are often more vulnerable and susceptible to being disrupted by enemy attack or weather. As such, it is useful to have a communications network linking these outposts to each other and to the rest of the network (headquarters, outposts that are not as vulnerable, etc.) that is capable of surviving the loss of a certain number of lines without being disconnected, as well as being capable of surviving the loss of vulnerable outposts and still have the remaining network able to survive the loss of a certain number of additional lines without being disconnected.

Although it has important applications, the problem we have just been describing has not hitherto been examined. In this dissertation, we introduce this new and important network design problem, calling it the *vital core connectivity problem* (VCC). More specifically, VCC is as follows: Let $G = (V, E)$ be a complete graph, let $k, l \in \mathbb{Z}$, $1 \leq l \leq k$, be non-negative integers, and let the vertices of G be partitioned into two sets – vital vertices V^*, and secondary vertices $V \backslash V^*$. The problem of VCC is to find a minimum cost subgraph such that there are at least k-edge disjoint paths between every pair of distinct vertices in G, and that if some or all of the secondary vertices are lost there remains in the graph restricted to the vital vertices at least l-edge disjoint paths between every pair of distinct vital vertices. Multiple copies of edges can be included in the solution. We denote a particular instance of VCC by VCC(k, l, V^*). 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 and provide the first approximation algorithm and integrality gap results for it.

1.1.1 Contributions

The major contributions of this dissertation are as follows:

1. We introduce a new network connectivity problem – the vital core connectivity problem (VCC). We show that this is an NP-hard problem. Despite its many practical uses, this problem has not been previously studied. We present the first constant factor approximation algorithm for VCC, with an overall approximation guarantee of at most $\frac{19}{6}$. We also provide an upper bound of at most $\frac{19}{6}$ on the integrality gap of the LP relaxation of VCC. We present further improved results for special cases of VCC.

2. We introduce a useful generalization of the minimum spanning tree problem (MST), called the *extended minimum spanning tree problem* (EMST), which is based on the special case of VCC for $k = l = 1$. We provide a polynomial-time algorithm that solves EMST, and provide a complete linear description for that problem. Furthermore, we show how to generalize it to handle numerous disjoint vital cores, providing the first complete linear description of, and polynomial-time algorithm for, the *vital core minimum spanning tree problem* (VCMST).

3. We consider multi-SNDP, and present a new approximation algorithm for which the approximation guarantee is better than the current best known approximation guarantee for certain cases of the problem. We also obtain improved integrality gap bounds with our method. We show the application of these results to variations of this problem.

4. We consider multi-SNDP, as well as VC-SNDP, with low vertex connectivity requirements. We investigate cases where the optimal values of multi-SNDP and SNDP are equal. We show that there is an upper bound of $\frac{3}{2}$ on the integrality gaps of the LP relaxations of both metric SNDP with vertex connectivity requirements 0,1,2, and of metric VC-SNDP with vertex connectivity

requirements 0,1,2. We also present $\frac{3}{2}$-approximation algorithms for both these problems. These results are an improvement on the previously best known integrality gap bound and approximation guarantee for the problems, which was 2 [20]. Our $\frac{3}{2}$ approximation guarantee and integrality gap bound applies to the Steiner tree problem with either uniform or metric edge costs, and improves the previous best known integrality gap bound for these cases.

5. The quality of the results that one can obtain for a given network design problem often depends on its ILP formulation, and, in particular, on its LP relaxation. In this connection, we investigate formulations for the Steiner tree problem (ST). We propose two new formulations for ST and investigate their strength in terms of their associated integrality gaps.

1.1.2 Outline

This dissertation is paper-based. Chapters 2, 3, 4, and 5 consist of individual papers, each of which stands completely on its own. A list of references is found at the end of each chapter, including at the end of Chapter 1 (Introduction) and Chapter 6 (Conclusion). An overview of this dissertation is as follows:

The remainder of Chapter 1 provides the general background for this dissertation. We provide an introduction and examination of the concepts of NP-hard, approximation algorithms, and integrality gaps. We also present notation and terminology that will be used throughout this dissertation. Finally, we examine each of the network design problems we address in this dissertation. For each of them, we provide a formal definition, special cases, and different versions, as well as a brief literature review for the problem and its different versions.

In Chapter 2, we introduce the new vital core connectivity problem (VCC). 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 the special case of VCC in which $k = l = 1$, which we call the *extended minimum spanning tree problem* (EMST). In addition, we provide a complete linear description for EMST. This paper is the first to introduce EMST, as well as the first to solve it. Both the polynomial-time algorithm and polyhedral description of EMST are fundamental to obtaining our results for VCC. We present the first constant factor approximation algorithm for VCC, as well as the first upper bound on the integrality gap of the LP relaxation of VCC. In particular, we show an approximation guarantee (and upper bound on the integrality gap) of $\frac{8}{3}$ for $l \geq \lceil \frac{k}{2} \rceil$; of $\frac{19}{6}$ for $2 \leq l < \lceil \frac{k}{2} \rceil$; and of $\frac{5}{2}$ 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. We note our approximation guarantee (and upper bound on the integrality gap) for several special cases. For example, for the special case VCC(k, k, V^*), $V^* \neq V$, we get 2; for the special case VCC$(2, 1, V^*)$ (in which the secondary vertices are not a cut set), we have $\frac{3}{2}$. Notice that when there is only one secondary vertex $r \in V$, VCC$(2, 1, V \backslash \{r\})$ has a $\frac{3}{2}$ approximation guarantee (and upper bound on the integrality gap) and finds a feasible multi-2EC (the special case of multi-SNDP with vertex connectivity requirements $r_i = 2$ for all $i \in V$) solution in which r is not a cut vertex. This extends our $\frac{3}{2}$-approximation algorithm for multi-2EC in Chapter 4. We present further improvements for the approximation guarantee and upper bound of the integrality gap for certain special cases. We also provide the first complete linear description of, and polynomial-time algorithm for, an extension of EMST to numerous disjoint vital cores.

In Chapter 3, we examine the NP-hard problem multi-SNDP, and present results which are improvements over the current best known results for certain cases of it. We begin by presenting some required background information and results on minimum cost perfect matchings, T-joins, and minimum cost spanning trees. Using such results, we show that the integrality gap for the LP relaxation of multi-SNDP is at most $\frac{3}{2}\frac{r_m}{r_l}$ when r_m is even, and at most $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ when r_m is odd, where r_m is the maximum vertex connectivity requirement and r_l is the smallest nonzero vertex connectivity requirement. We present an approximation algorithm for multi-SNDP which has an approximation guarantee the same as these upper bounds. These are improvements for the best known bounds and approximation guarantees for certain special cases of multi-SNDP, and we presented a family of examples demonstrating this. We also show the applications of our results to various versions and special cases of SNDP, and discuss the relationship of our approximation algorithm and the improved tree heuristic of Goemans and Bertsimas [28].

In Chapter 4, we examine the special cases of both the survivable network design problem and the vertex connected survivable network design problem (and related variations of these problems) with low vertex connectivity requirements. We demonstrate that multi-SNDP and metric SNDP are closely related for $r_i \in \{0, 1, 2\}$, $i \in V$; and we show that the upper bound for the integrality gap of the LP relaxation of metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, is $3/2$, thus extending the known upper bound of $3/2$ for the integrality gap of the LP relaxation of metric 2EC [4]. We also show that our multi-SNDP approximation algorithm (presented in Chapter 4), with certain modifications, applies as well for metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, thus giving a $\frac{3}{2}$-approximation algorithm for that case. We investigate VC-SNDP, particularly for small vertex connectivity requirements. We further modify our approximation algorithm so that we get a $\frac{3}{2}$-approximation algorithm for metric VC-SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$. We also show there is a $\frac{3}{2}$ upper bound for the integrality gap of the LP relaxation of this problem. Our $\frac{3}{2}$ approximation guarantee and integrality gap bound applies to the Steiner tree problem with either uniform or metric edge costs. Restricted to metric 2VC (the problem, given metric edge costs, of finding a minimum cost subgraph that has at least two internally vertex-disjoint paths between every pair of distinct vertices), our algorithm (but not the proof) is similar to the $\frac{3}{2}$-approximation algorithm for metric 2VC given by Frederickson and Ja'Ja' [21], and thus our proof can provide an alternate, more compact proof of their result.

Finally, in Chapter 5 we investigate different ILP formulations for ST, and their corresponding LP relaxations. The integrality gap of the LP relaxation of any network problem depends on the formulation of the linear program – one LP formulation for a given problem may give a smaller integrality gap than another LP formulation for the same problem. Since ST is NP-hard, it is considered highly unlikely that we can get an exact LP relaxation (i.e., with integrality gap 1) for ST like we can for its special case MST. Our goal, then, is to obtain the best possible LP relaxation for ST; in other words, to find an LP relaxation for ST that has an integrality gap as close to 1 as possible. In particular, in this chapter we consider finding LP relaxations for ST such that the LP relaxation, when restricted to MST, has an integrality gap of 1. Such an LP relaxation for ST may have an integrality gap closer to 1 and thus be able to be used to obtain stronger results for ST. We examine some known ILP formulations for ST, and the bounds on the integrality gap of their LP relaxations. We then propose a new ILP formulation for ST that is based on the well-known undirected cut formulation. However, for the special case of MST, the LP relaxation of this proposed formulation does not have an integrality gap of 1, and thus the formulation does not look promising to use in obtaining improved results for ST. We propose a second new ILP formulation for ST, this one based on Steiner partitions. For the special case of MST, the LP relaxation of this formulation does not have an integrality gap of 1; however, it is 1 if we include a broader set of constraints. We present a family of graphs that provides a lower bound of $8/5 = 1.6$ for the integrality gap of the LP relaxation of the proposed formulation.

We conclude this dissertation in Chapter 6 by summarizing the results that are presented in it, and by presenting some recommendations and possible avenues for future research.

1.2 Background and Notation

1.2.1 NP-hard, Approximation Algorithms, and Integrality Gap

A *decision problem* is a problem whose answer is "Yes" or "No." Every optimization problem has a corresponding decision problem. For example, a well-known optimization and network design problem is the minimum cost 2-edge connected subgraph problem: Given a graph which has an edge between every pair of distinct vertices, and a non-negative cost associated with each edge for building that edge, find a cheapest set of edges to construct such that an edge can be lost without disconnecting the network. The decision form of this problem (which has a "Yes" or "No" answer) is as follows: Given a graph which has an edge between every pair of distinct vertices and a non-negative cost associated with each edge for choosing that edge, and given a number C, is there a set of edges such that if one of them is lost the network is still connected and such that it has total cost less than or equal to C?

The class $\mathcal{P}$ (*polynomial*) is the set of decision problems that can be solved in polynomial time (essentially, "efficiently" and "quickly"). The class $\mathcal{NP}$ (*non-deterministic polynomial*) is the set of decision problems such that if the answer is "Yes", then a proof of that answer can be verified in polynomial time; i.e., it is the set of decision problems such that, given a potential solution, we can verify in polynomial time whether or not that candidate is in fact a solution to the problem [15]. Every problem in $\mathcal{P}$ is also in $\mathcal{NP}$, i.e. $\mathcal{P} \subseteq \mathcal{NP}$; however it is a long-standing open problem to determine if $\mathcal{NP}$ consists solely of problems in $\mathcal{P}$, or if it in addition contains other problems which are not part of $\mathcal{P}$. Succinctly, it is an open problem to determine whether $\mathcal{P} = \mathcal{NP}$ [2]. It is conjectured and generally thought that $\mathcal{P} \neq \mathcal{NP}$. In

[2]Essentially, this would mean that if a solution can be verified in polynomial time, then a solution can also be found in polynomial time [18].

fact, the Clay Mathematics Institute (CMI) of Cambridge, Massachusetts, is offering a $1,000,000 prize to anyone who can prove (or disprove) this conjecture [14].

A decision problem B that is in the class $\mathcal{NP}$ is *polynomial-time reducible* to a decision problem A that is also in $\mathcal{NP}$ if there exists a polynomial time algorithm which transforms an instance of problem B to an instance of problem A such that the answer to the instance of B is "Yes" if and only if the answer to the instance of A is "Yes" [15, 40]. A decision problem A is *NP-complete* if it is in $\mathcal{NP}$ and every problem in $\mathcal{NP}$ is polynomial time reducible to A [15, 40]. Notice that this implies that if one NP-complete problem can be proved to be solvable in polynomial time (i.e., that it is in $\mathcal{P}$), then every problem in $\mathcal{NP}$ is in $\mathcal{P}$, and thus $\mathcal{P} = \mathcal{NP}$. To prove that a problem A in $\mathcal{NP}$ is a NP-complete problem, it is sufficient to show that an already known NP-complete problem is polynomial-time reducible to A.

Given an optimization (or decision) problem P, an algorithm has *oracle access* to P if it can solve an instance of P with a single instruction [63]. An optimization (or decision) problem P is *NP-hard* if there is a polynomial time algorithm for an NP-complete problem A when the algorithm has oracle access to P [63]. The term "NP-hard", although defined in general, is most frequently applied to an optimization problem whose corresponding decision form is NP-complete [30, 63]. It follows that if there exists a polynomial-time algorithm for finding an optimal solution to any NP-hard problem, then $\mathcal{P} = \mathcal{NP}$ (simply solve the corresponding decision problem by running the polynomial time algorithm on an instance of the decision problem and verify the solution's total cost, thus solving the NP-complete problem in polynomial-time and showing it to be in $\mathcal{P}$). Figure 1.1 illustrates the relationship between $\mathcal{P}$, $\mathcal{NP}$, NP-complete, and NP-hard problems [58]. An early compendium of NP problems is given in [26], and a compendium of NP-hard problems up until 1998 is given in [16].

We will show that many of the network design problems that we consider are NP-hard. Since it is conjectured that $P \neq NP$, it is considered highly unlikely that

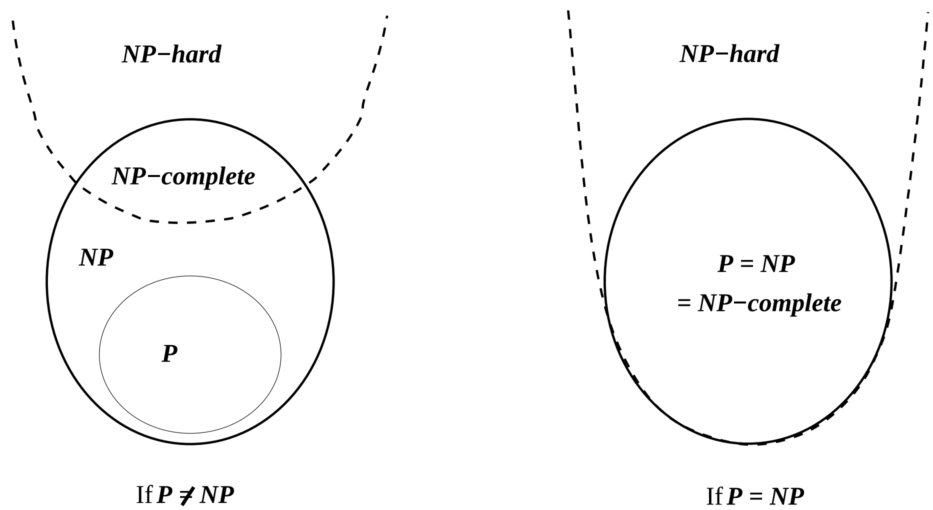

Figure 1.1: Relationship of $\mathcal{P}$, $\mathcal{NP}$, NP-complete, and NP-hard problems [58].

efficient algorithms for exactly solving NP-hard problems exist. We thus look at designing efficient algorithms that return approximate, or near-optimal, solutions for NP-hard problems. Given a problem, if an algorithm runs in polynomial time and returns a solution whose value is always within a constant factor $\alpha \geq 1$ of the optimal value, i.e.,

$$(\textit{cost of algorithm solution}) \leq \ \alpha \cdot (\textit{cost of optimal solution}),$$

then the algorithm is known as a *(constant factor)* α-*approximation algorithm*, and the factor α is called the *approximation* or *performance guarantee* of the algorithm.

In this dissertation, we deal with many different minimization problems on weighted graphs (i.e., graphs with edge costs), in which we seek to find a set of edges of the graph that satisfy a set of given properties and have minimum total cost. Given such a minimization problem P, and a graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, $c \geq 0$, let x_e represent the number of times each edge $e \in E$ is included in a feasible solution of P on G. The vector $x \in \mathbb{N}^E_{\geq 0}$ is called an *incidence vector* on G. An *integer linear programming* (ILP) *formulation* of P minimizes the total edge cost over all the feasible solutions to P, and typically is of the form:

$$\begin{array}{rrcll} \text{minimize} & cx & & & \\ \text{subject to} & ax & \geq & b_1, & \\ & dx & \leq & b_2, & \\ & fx & = & b_3, & \\ & x_e & \geq & 0, & \text{for all } e \in E, \\ & x_e & & \text{integer}, & \text{for all } e \in E, \end{array}$$

where $a, d, f \in \mathbb{R}^E$ and $b_1, b_2, b_3 \in \mathbb{R}$. For a solution $x' \in \mathbb{N}^E_{\geq 0}$ to an integer linear programming formulation of such a minimization problem on G, the corresponding subgraph, H, of G is obtained by taking x'_e copies of edges of G, for all $e \in E$. A *linear programming* (LP) *formulation* of P is a linear formulation in which the values of the variables x_e do not have to be integer.

An important and related concept to approximation guarantees is that of integrality gaps. Given a minimization problem P, and its integer linear program (ILP) Q, a linear program (LP) is a *linear programming relaxation*, Q_{LP}, of Q if every feasible solution of Q is also a feasible solution of Q_{LP}. In this dissertation, we refer to the LP relaxation of an ILP as the linear programming relaxation formed by relaxing (i.e., removing) the integrality constraints in the ILP. In such a case, every integer solution of Q_{LP} is a solution of Q. Let $opt(Q)$ be the value of an optimal solution to Q, and let $opt(Q_{LP})$ be the value of an optimal solution to Q_{LP}. The optimal value of Q_{LP} is at most the optimal value of Q, and thus provides a lower bound on the optimal value of Q. The question naturally arises as to how big a difference, or "gap," can possibly exist between $opt(Q)$ and $opt(Q_{LP})$. This is where the notion of integrality gap comes in.

Formally, the *integrality gap* of a linear programming relaxation of a minimization problem P is the largest ratio $opt(Q)/opt(Q_{LP})$ over all possible non-negative, and all not identically 0, objective cost functions. If it can be proved that this largest ratio has a value β (where β is some constant value), the linear programming relaxation is said to have a β *integrality gap*; indicating that, in the worst possible case, $opt(Q)$ is no more than $\beta \cdot opt(Q_{LP})$. Thus, the integrality gap provides a measure of the quality of the lower bound provided by Q_{LP} for Q. If the integrality gap of a problem's LP relaxation is 1, then that LP relaxation is said to be *exact*, and there is always an optimal solution of that LP formulation that is integer and thus an optimal solution for Q.

It is important to note that the integrality gap of a LP relaxation depends on the formulation of the linear program – one formulation for a given problem may give a better integrality gap than a different formulation for the same problem. This can be seen simply from polyhedral theory[3]: Since the underlying polyhedron of Q_{LP} contains the underlying polyhedron of Q, taking a facet of the polyhedron associated with Q and adding it as a constraint to Q_{LP} reduces the feasible area of the LP relaxation, thus giving a stronger LP relaxation and potentially reducing the difference, or "gap", between the value of an optimal solution to Q and the value of an optimal solution to the new LP relaxation. Given a specific graph $\hat{G}$ with non-negative edge costs, we refer to the *integrality ratio* as the ratio of the cost of an optimal solution to a given ILP on $\hat{G}$ over the cost of an optimal solution to the LP relaxation of that ILP. For a given ILP, every integrality ratio gives a lower bound on the corresponding integrality gap of the LP relaxation.

Approximation guarantees and integrality gaps, although distinct concepts, are closely related: A polynomial-time constructive proof for a bound $\beta \geq 1$ on the integrality gap of an LP relaxation Q_{LP} forms an β-approximation algorithm for problem P. In other words, if, for all graphs, we can construct, in polynomial-time, a feasible solution H (with total edge cost $cost(H)$) to Q such that $cost(H) \leq$

[3]For terminology and concepts of polyhedral theory, see [15]

β $opt(Q_{LP})$, then we have both

$$opt(Q) \leq cost(H) \leq \beta\ opt(Q_{LP});\ \ \text{i.e.,}\ \frac{opt(Q)}{opt(Q_{LP})} \leq \beta,$$

and

$$cost(H) \leq \beta\ opt(Q_{LP}) \leq \beta\ opt(Q);\ \ \text{i.e.,}\ \frac{cost(H)}{opt(Q)} \leq \beta,$$

and hence we have an upper bound of β for the integrality gap for the LP relaxation Q_{LP}, as well as a β-approximation algorithm for the problem P. In addition to this connection, many approximation algorithms are based on an LP relaxation, with the algorithm returning a feasible ILP solution whose cost is close to the cost of an optimal LP solution [44] (for example, the primal-dual method). Thus, obtaining an LP relaxation that has a small (close to 1) integrality gap, can aid in obtaining better approximation algorithms for P. In fact, it has been observed that often the best known approximation guarantee and the best known lower bound for the integrality gap for a problem match [10].

1.2.2 Terminology and Notation

This section presents the notation that will be used throughout this dissertation.

Let G be a graph, with vertex set $V(G)$ and edge set $E(G)$. Such a graph G is denoted $G = (V, E)$, where $V = V(G)$ and $E = E(G)$. The number of vertices in the set V is denoted $|V|$, and the number of edges in the set E is denoted $|E|$. A graph G is a *complete graph* if it has an edge between every pair of distinct vertices. Besides being referred to directly by name, an edge in a graph can also be referred to by its endpoints. In particular, if $e \in E$ joins the vertices $u, v \in V$, $u \neq v$, then e can also be written as uv (meaning the [undirected] edge joining the vertices u and v).

Given a graph $G = (V, E)$, for $S \subseteq V$, let $\delta_G(S)$ denote the set of edges $\{uv \in E : u \in S, v \notin S\}$, i.e, the set of edges of G with exactly one end in S. The set $\delta_G(S)$ is called a *cut*. Let $\gamma_G(S)$ be the set of edges $\{uv \in E : u, v \in S\}$, i.e., the set of edges of G with both ends in S. The subscript G will be left out when it is clear to which graph we are referring.

When there are no costs (or weights) on the edges of G, G is called an *unweighted* graph. When the edge costs $c \in \mathbb{R}^E$ of a graph $G = (V, E)$ are non-negative and satisfy the triangle inequality $c_{uw} \leq c_{uv} + c_{vw}$ for all distinct vertices $u, v, w \in V$, the costs are called *metric*. Given a complete graph $G = (V, E)$ with non-negative edge costs c, the *metric completion*, $G^{\triangle} = (V, E)$, of G is obtained by setting the edge cost $c^{\triangle}_{ij}$ to be that of a minimum-cost path connecting vertices i and j in G, for every edge $ij \in E$. Clearly, the edge costs $c^{\triangle}$ are metric.

A *multi-graph* is a graph that, for at least one pair of distinct vertices, has multiple copies of edges between that pair of vertices[4]. Such multiple copies of edges are considered to be distinct edges, and may have distinct edge costs. On the other hand, when there are no multiple copies of edges in G, G is *simple*.

A *subgraph* of a graph $G = (V, E)$ consists of a set of vertices V' that are a subset of V, and a set of edges E' that are a subset of E and for which both endpoints lie in V'; i.e., the graph (V', E') is a subgraph of G if $V' \subseteq V$ and $E' \subseteq \gamma_G(V')$ $(\subseteq E)$. A *spanning subgraph* of G is a subgraph of G that includes all the vertices V of G (i.e., it is a graph (V, E') where $E' \subseteq E$). Let $G[V']$ be the subgraph of G induced by the set of vertices V'. In other words, $G[V']$ consists of the vertices V' and the edges $\gamma_G(V')$.

A *tree*, T, of a graph $G = (V, E)$ is a subgraph of G that is connected and does not contain any cycles. If the vertices of T are all of V, then T is a *spanning tree* of G.

A *path* from a vertex u to a vertex v in a graph G is an ordered sequence of distinct vertices in G in which each adjacent pair of vertices is linked by an edge. A graph G is *connected* if, for each vertex, there is a sequence of consecutive edges from

[4] A multi-graph does not allow loops.

that vertex to every other vertex, i.e., there is a path between every pair of distinct vertices. A set of paths between vertices i and j is pairwise *internally vertex-disjoint* if no two of the paths have an internal vertex in common (i.e., the only vertices any of the paths have in common are the vertices i, j). A path that starts and ends at the same vertex is called a *cycle*. 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 distinct vertices ($k \in \mathbb{N}_{\geq 1}$); i.e. if any $k-1$ edges (respectively, vertices) are removed, the graph is still connected.

If a graph $G = (V, E)$ is k-edge connected, then, for every $S \subset V$, the cut $\delta(S)$ contains at least k edges; in particular, the degree of each of its vertices is at least k. As well, if a graph is k-edge connected, then that graph is j-edge connected for $1 \leq j < k$. A connected graph with n vertices and m edges (thus $m \geq n-1$) has edge connectivity at most $\lceil \frac{2m}{n} \rceil$ [34]. The trivial graph (consisting of a single vertex) is defined to be k-edge connected, $k \in \mathbb{N}_{\geq 1}$.

A *cut vertex* in a connected graph is a vertex whose removal disconnects the graph. A graph is 2-vertex connected if and only if it has no cut vertices. In a k-vertex connected graph, a vertex *cut set* is a subset $U \subset V$ whose removal disconnects the graph (and such that $|U| \leq k-1$). If a graph is k-vertex connected, then it is j-vertex connected for $1 \leq j < k$, $j \in \mathbb{N}_{\geq 1}$, and it has at least $k+1$ vertices. A complete graph with n vertices is always $(n-1)$-vertex connected [34]. It can be shown that a connected graph with n vertices and m edges ($n, m \in \mathbb{N}_{\geq 1}$, $m \geq n-1$) has vertex connectivity at least $\max\{m - \frac{(n-1)(n-2)}{2}, 0\}$ and at most $\lceil \frac{2m}{n} \rceil$ [34]. All k-vertex connected graphs are k-edge connected, but not vice versa. See Figure 1.2.

Given a minimization problem $\mathcal{P}$ defined on the edge set E of a graph $G = (V, E)$, its integer linear program $\mathcal{Q}$, and its linear programming relaxation $\mathcal{Q}_{LP}$, let $opt_G(\mathcal{Q})$ be the total edge cost of an optimal solution to the integer linear programming problem $\mathcal{Q}$ on G, and let $opt_G(\mathcal{Q}_{LP})$ be the cost of an optimal solution to $\mathcal{Q}_{LP}$ on G. Given a solution H to problem $\mathcal{P}$ on $G = (V, E)$, let x_e represent the number of times

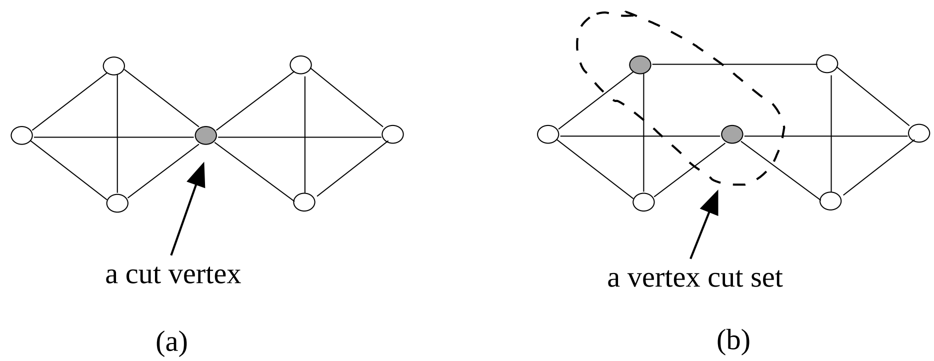

Figure 1.2: (a) A graph that is 3-edge connected, but only 1-vertex connected (since it has a cut vertex). (b) A graph that is 3-edge connected, 2-vertex connected (it is not 3-vertex connected since it has some cut sets of size 2).

each edge $e \in E$ is included in the solution. For any edge set $E' \subseteq E$ and $x \in \mathbb{R}^E$, let $x(E')$ denote the sum $\sum_{e \in E'} x_e$. Let $cost_G(x') := \sum_{e \in E} c_e x'_e$ be the total edge cost of a feasible solution x' to problem $\mathcal{Q}$; and let $cost_G(\mathcal{H})$ be the total edge cost of $\mathcal{H}$, where $\mathcal{H}$ is either a subgraph of G or a subset of edges of E. The subscript G will be omitted when it is clear which graph we are discussing.

For terminology and concepts of polyhedral theory, we refer the reader to [15].

1.2.3 Network Connectivity Problems

In this section, we introduce various network connectivity problems, some of which are well-known, and some of which are newly presented in the chapters of this dissertation. In one way or another, this dissertation involves each of them – whether it be using them in solutions to other problems, or obtaining further results for them, or presenting approximation algorithms with improved approximation guarantees, or finding improved bounds on integrality gaps. Notice that for most of these problems, a complete graph is assumed. This can be accomplished without loss of generality for weighted graphs (i.e., graphs with edge costs), since, if G is not complete, we can make it into a complete graph by simply adding in all the "missing" edges and assigning each of them an appropriately large cost.

Spanning Tree Problem

Given a connected graph G with costs on its edges, the *minimum cost spanning tree problem* (MST) is the well-known problem of finding a spanning tree of G that has the lowest cost. This problem is one of the few network connectivity to an optimal solution by using Prim's or Kruskal's polynomial-time algorithms [9, 15].

2-edge Connected and 2-vertex Connected Spanning Subgraph Problems

Given a complete graph G with non-negative edge costs $c \in \mathbb{R}^E$, the *minimum cost 2-edge connected spanning subgraph problem* (2EC) is to find a minimum cost spanning subgraph of G that is 2-edge connected. If the graph is unweighted, or, equivalently, the edge costs are all 1 (or, equivalently, the edge costs are all uniform), the problem is to find a *minimum size* (or *minimum cardinality*) subgraph of G (i.e., a subgraph containing the minimum number of edges possible) that is 2-edge connected (denoted *min-size 2EC*). Since 2EC is a special case of all the network connectivity problems we examine (except MST), we examine 2EC, and previously known results for it, in a little more depth. When G has metric edge costs, 2EC will be referred to as *metric 2EC*. When multiple copies of edges are allowed in the solution of 2EC, the problem will be referred to as *multi-2EC*.

Similarly, given a complete graph G with non-negative edge costs $c \in \mathbb{R}^E$, the *minimum cost 2-vertex connected spanning subgraph problem* (2VC) is to find a minimum cost spanning subgraph of G that is 2-vertex connected. If the graph is unweighted, the problem is to find a minimum size (or minimum cardinality) subgraph of G that is 2-vertex connected (denoted *min-size 2VC*). When G has metric edge costs, 2VC will be referred to as *metric 2VC*.

It can be shown that the optimal values of metric 2EC and metric 2VC are equal. This is an implicit result of the shortcutting method by Frederickson and Ja'Ja' [21], and is used by Carr and Ravi [9], and by Monma, Munson, and Pulleyblank [51].

Problems 2EC and 2VC are special cases of, among other problems, the *graph augmentation problem*: Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, and a subgraph (V, F), $F \subset E$, find a set of edges $E' \subseteq E \backslash F$ of minimum cost such that the "augmented graph", $(V, F \cup E')$, satisfies a given property [21]. The graph augmentation problem can be used for improving the connectivity of already existing networks. For 2EC, $F = \emptyset$ and the property to be satisfied is that the augmented graph $(V, F \cup E') = (V, E')$ must be two-edge connected. For 2VC, $F = \emptyset$ and the property to be satisfied is that the augmented graph $(V, F \cup E') = (V, E')$ must be two-vertex connected. For this reason, 2EC is sometimes called the *bridge-connected augmentation problem*, and 2VC is sometimes called the *biconnected augmentation problem*. Merging the problems together, we obtain the *bi-level augmentation problem*, in which, for given $U_1, U_2 \subseteq V$, we want to add edges in a minimum cost or minimum size way to make the induced subgraph $G[U_1]$ 2-vertex connected, and the induced subgraph $G[U_2]$ 2-edge connected. Clearly, 2EC and 2VC are both special cases of this problem; with $U_1 = \emptyset$, $U_2 = V$, and $U_1 = V$, $U_2 = \emptyset$, respectively.

If G is a complete graph with unweighted edges, then the bridge-connected and bi-connected augmentation problems (and hence min-size 2EC) can be solved exactly by efficient (i.e., polynomial time) algorithms. Such an algorithm was presented by Eswaran and Tarjan in 1976 [19]. In 1998, Hsu and Kao went further and presented an algorithm that solves the unweighted bi-level augmentation problem (on a complete graph) in linear time [36]. If, on the other hand, the problems are to be solved on a general graph, or if the edges have different edge costs, then the augmentation problems are NP-hard [16, 21]. In particular, min-size 2EC is NP-hard [43]. This can be seen since the NP-complete Hamiltonian cycle problem (that of finding a cycle that traverses all the vertices of the graph exactly once) reduces to it[5]. Since min-size 2EC is a special case of 2EC, 2EC is also NP-hard. Similarly, min-size 2VC, and thus 2VC, is NP-hard [43, 51].

Initially, a 2-approximation algorithm was known for min-size 2EC. Khuller and Vishkin were the first to improve on it in 1992, when they gave a $\frac{3}{2}$-approximation algorithm based on depth-first search (they also presented a $\frac{5}{3}$-approximation algorithm for min-size 2VC) [43]. In 1998, Cheriyan, Sebö and Szigeti provided a $\frac{17}{12}$-approximation algorithm [12]. This approximation algorithm also formed a constructive proof that the integrality gap of the LP relaxation of min-size 2EC is at most $\frac{17}{12}$.

In 2000, Vempala and Vetta improved on the $\frac{17}{12}$-approximation guarantee, presenting $\frac{4}{3}$-approximation algorithms for both min-size 2EC and min-size 2VC. The running time of the algorithm is $O(n^2m)$, where m is the number of edges in the graph, and n the number of vertices [62]. This $\frac{4}{3}$-approximation algorithm for min-size 2EC is significant, since an unverified implication of the $\frac{4}{3}$-TSP conjecture[6] in combinatorial optimization is that 2EC is approximable to within $\frac{4}{3}$. Since min-size 2EC is a special case of 2EC, Vempala and Vetta's $\frac{4}{3}$-approximation algorithm partially verifies this, and thus lends support to the $\frac{4}{3}$-TSP conjecture.

Regarding 2VC, in 1982 Frederickson and Ja'Ja' examined the biconnected augmentation problem on complete graphs with metric edge costs [21]. In their significant paper, they presented a $\frac{3}{2}$-approximation algorithm for metric 2VC.

The best approximation guarantee for 2EC and multi-2EC is 2. This guarantee is from Jain, who, in 2001, presented a 2-approximation algorithm for finding

[5]Given an unweighted complete graph $G = (V, E)$, if min-size 2EC returns $|V|$ as the minimum number of edges, then G has a Hamilton cycle, otherwise, it does not.

[6]The famous *travelling salesman problem* (TSP) is the problem of finding a minimum cost Hamilton cycle. The $\frac{4}{3}$-TSP conjecture says that, for metric costs, the integrality gap of the LP relaxation of TSP has an upper bound of 4/3.

a minimum-cost subgraph (of a multi-graph) having at least a specified number of edges in each cut [37]. Problem 2EC is a special case of this problem. In addition to providing an approximation guarantee, Jain's approximation algorithm gives a constructive proof that the integrality gap of the integer relaxation of his problem is 2, and at most 2 for all subclasses of the problem (such as 2EC). He also provides an asymptotically tight example showing the integrality gap of the general class is 2. In particular, Jain's result implies that there is an upper bound of 2 on the integrality gaps of the LP relaxations of 2EC and multi-2EC.

In 2006, Alexander, Boyd, and Elliott-Magwood proved that the integrality gap for the LP relaxation of multi-$2EC$ is less than or equal to $3/2$ [4].

The best approximation guarantee for 2VC is 2 [47].

k-edge Connected and k-vertex Connected Spanning Subgraph Problems

Given a complete graph, $G = (V, E)$, with non-negative edge costs $c \in \mathbb{R}^E$, the *k-edge connected subgraph problem* (kEC), $k \in \mathbb{N}_{\geq 1}$, is the problem of finding a minimum cost k-edge connected spanning subgraph of G. Notice that 1EC is precisely MST. A kEC problem with metric edge costs will be referred to as *metric kEC*. If multiple copies of the edges are allowed in the solution of kEC, the problem will be referred to as the *multi-k-edge connected subgraph problem* (multi-kEC). Both kEC and multi-kEC ($k \geq 2$) are known to be NP-hard problems [26, 46].

Similarly, given a complete graph, $G = (V, E)$, with non-negative edge costs $c \in \mathbb{R}^E$, the *k-vertex connected subgraph problem* (kVC) is the problem of finding a minimum cost k-vertex connected spanning subgraph of G. When multiple copies of edges are allowed in the solution, we denote the problem by *multi-kVC*. Problem kVC with metric edge costs will be referred to as *metric kVC*. Since adding multiple copies of edges does not increase the vertex connectivity, kVC with multiple copies of edges in its solution is generally not considered. Problem kVC, generally for low values of k, is of particular interest for wireless networks, where the towers/centres can fail, but the wireless "links" do not [39]. It is known that kVC is a NP-hard problem [45].

For 3EC with G either a simple or a multi-graph and uniform edge costs (equivalently, unweighted), there is a 4/3-approximation algorithm [33]. Concerning metric 2EC, there is a known upper bound of 3/2 for the integrality gap of the LP relaxation of metric 2EC [4] and a $\frac{3}{2}$-approximation algorithm for metric 2EC [21]. For kEC on a multi-graph with uniform edge costs, there is a 1.61 approximation algorithm [22]. For multi-kEC with uniform edge costs, a $1 + \frac{2}{k}$ approximation algorithm exists, as well as an upper bound of $1 + \frac{2}{k}$ on the integrality gap of the LP relaxation of the problem [23]. For kEC and multi-kEC, there is a 2-approximation algorithm as well as an upper bound of 2 on the integrality gap of the respective LP relaxations [37]. For a summary of current results on the different versions of kEC, see [46].

Problems involving vertex connectivity are generally harder problems for which to obtain results than are problems involving edge connectivity. For a summary of the current known approximation guarantees on the different versions of kVC, see [46]. Currently, the best approximation guarantee for kVC, $k \geq 2$, is a k-approximation algorithm [47]. There is also a 2-approximation algorithm for 3VC [6] and a 3-approximation algorithm for 4VC and 5VC [17]. For metric kVC, a $(2 + (k-1)/n)$-approximation algorithm exists, where $n = |V|$ [47]. For metric 2VC, there is a $\frac{3}{2}$-approximation algorithm [21]. A (non-constant) upper bound for the integrality gap of the LP relaxation of kVC is given in [48]. When edge costs are all 1 (i.e., uniform edge costs, or, equivalently, on an unweighted graph), there is a 4/3-approximation algorithm for 2VC [62].

Edge-connected & Vertex-connected Survivable Network Design Problem

The *(edge connected) survivable network design problem* (SNDP) (also called the *generalized Steiner network problem*) is as follows: Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. Vertices $i \in V$ with vertex connectivity requirement $r_i = 0$ are called *Steiner vertices*. For each pair of distinct vertices $i, j \in V$, let $r_{ij} = \min\{r_i, r_j\}$. The problem is to find a minimum cost subgraph that has, for each pair of distinct vertices $i, j \in V$, at least r_{ij} edge-disjoint paths between i and j. In a feasible solution of SNDP, the loss of any $0 < l < \max_{ij \in E} r_{ij}$ edges still allows all vertices with vertex connectivity requirement greater than l to be connected to each other (i.e., for $i, j \in V$, if $r_i > l$ and $r_j > l$, then if any l edges are removed there remains at least one edge-disjoint path joining i and j).

If multiple copies of the edges are allowed in the solution of SNDP, the problem will be referred to as the *multi-survivable network design problem* (multi-SNDP). The equivalent cut requirement version of multi-SNDP is: Given a cut requirement function $f : 2^V \rightarrow \mathbb{Z}$ defined on the subsets of V by $f(S) = \max_{i \in S,\ j \notin S} r_{ij}$ for all $S \subset V$, find a minimum cost multi-subgraph having at least $f(S)$ edges in each cut $\delta(S)$, $S \subset V$. Note that if G is not complete, we can make it into a complete graph by adding in all the "missing" edges and assigning each of them an appropriately large cost. When the edge costs $c \in \mathbb{R}^E$ of a graph are non-negative and satisfy the triangle inequality $c_{uw} \leq c_{uv} + c_{vw}$ for all distinct vertices $u, v, w \in V$, the costs are called *metric*. A SNDP problem with metric edge costs will be referred to as *metric SNDP*.

Problems kEC and multi-kEC are special cases of SNDP and multi-SNDP, respectively. Thus, SNDP and multi-SNDP are NP-hard problems (see also [26, 40]).

The *vertex connected survivable network design problem* (VC-SNDP) is as follows: Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. Find a minimum cost subgraph that has, for each pair of distinct vertices $i, j \in V$, at least r_{ij} internally vertex-disjoint paths between i and j, where $r_{ij} = \min\{r_i, r_j\}$. In such a feasible solution, the loss of any $0 < l < \max_{ij \in E} r_{ij}$ vertices still allows all vertices with vertex connectivity requirement greater than l to be connected with each other (i.e., for $i, j \in V$, if $r_i > l$ and $r_j > l$, then if any l vertices are removed there remains at least one internally vertex-disjoint path joining i and j). Note that all feasible solutions to VC-SNDP are also feasible solutions to SNDP, but not vice versa. Thus, the optimal value of VC-SNDP is always at least that of SNDP. A *cut vertex* in a connected graph is a vertex whose removal disconnects the graph. Generally, multiple copies of edges are not allowed in the solutions of VC-SNDP, since multiple edges do not increase the vertex connectivity. Clearly, kVC is the special case of VC-SNDP with $r_i = k$ for all $i \in V$. Since kVC is NP-hard [6], thus so is VC-SNDP.

A summary of some of the previously known results for SNDP is given in [1, 50]. The best general constant factor approximation guarantee that has existed for multi-SNDP and SNDP has, to this point, been 2. This guarantee is due to Jain [37], who, in 2001, presented a 2-approximation algorithm for finding, in a general graph that may not be simple and has non-negative edge costs, a minimum cost subgraph having at least a specified number of edges in each cut [37]. Initially, multiple copies of edges are not allowed in the solution, however, as is noted near the end of Jain's paper, the 2-approximation algorithm holds as well for the case when multiple copies of edges are allowed in the solution. In addition to providing an approximation guarantee, Jain's algorithm provides a constructive proof that the integrality gap of the linear programming relaxation of the general problem he considers is at most 2. In particular, Jain's result implies that the integrality gap of the LP relaxation of SNDP and multi-SNDP is at most 2.

In 1993, Goemans and Bertsimas [28] presented an approximation algorithm (the improved tree heuristic) for multi-SNDP that, given distinct vertex connectivity requirements $\rho_o = 0 < \rho_1 < \rho_2 < \ldots < \rho_q$, has approximation guarantee $\sum_{j=1}^{q} \frac{f(\rho_j - \rho_{j-1})}{\rho_j}$, where $f(c) = \frac{3}{2}c$ when c is even, and $f(c) = \frac{3c+1}{2}$ when c is odd [28]. The proof used to obtain this is rather complex and involved, and, although not indicated, implicitly yields an upper bound for integrality gap as well. As a special case, this approximation algorithm has an approximation guarantee (and integrality gap) for multi-kEC of $\frac{3}{2}$ when k is even, and $\frac{3}{2} + \frac{1}{2k}$ when k is odd.

Concerning VC-SNDP, there is a 2-approximation algorithm for VC-SNDP with vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$ [20]. This is the first constant factor approximation guarantee for VC-SNDP (with low vertex connectivity requirements). A non-constant approximation guarantee of $2^{\log^{1-\epsilon} n}$ (for any fixed $\epsilon > 0$, and for $n = |V|$) exists for VC-SNDP [45]; however, to date, no constant factor approximation algorithm exists for VC-SNDP [20]. For VC-SNDP with $r_i \in \{1, 2\}$, $i \in V$ on an unweighted graph, there is a 3/2-approximation algorithm [50]. For a short summary of some of the known results on VC-SNDP, see [50].

Steiner Tree Problem

Given a general graph, $G = (V, E)$, and a subset of vertices $T \subseteq V$, a *Steiner tree* is a tree that connects the vertices in T (and may also include some, all, or no vertices from $V \backslash T$). Vertices in T are called *terminal* (or *required*) *vertices*, and non-terminal vertices in $V \backslash T$ are called *Steiner vertices* [29, 44]. Given a general graph, $G = (V, E)$, with non-negative edge costs $c \in \mathbb{R}^E$, and a set of terminal (or required) vertices $T \subseteq V$, the *Steiner tree problem* (ST) (sometimes also called the *Steiner minimal tree*) is the problem of finding a minimum cost Steiner tree of G; i.e., a minimum cost tree that contains all the terminal vertices T and any subset (including the empty set) of the Steiner vertices. Equivalently, it is the problem of finding a minimum cost tree that spans the vertex set T [13]. A feasible solution to ST may include some, all, or no Steiner vertices. Problem ST is a special case of

SNDP and multi-SNDP. Problem ST is the special case of SNDP with $r_i \in \{0,1\}$ for all $i \in V$. Notice also that ST is a special case of Jain's general network problem [37].

Given a general graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \{0, k\}$ for all $i \in V$, $k \geq 1$, the *Steiner k-edge connected subgraph problem* (Steiner-kEC) is that of finding a minimum cost k-edge connected subgraph of G that spans the set of terminal vertices $S = \{i \in V \mid r_i = k\} \subseteq V$; i.e., such that there are at least k edge-disjoint paths between all pairs of distinct vertices $i, j \in S$. If G is a complete graph and multiple copies of the edges are allowed in the solution of Steiner kEC, the problem will be referred to as the *Steiner k-edge connected multi-subgraph problem* (Steiner multi-kEC). Problems Steiner kEC and Steiner multi-kEC are special cases of SNDP and multi-SNDP, respectively. Furthermore, problems kEC and multi-kEC are special cases of Steiner kEC and Steiner multi-kEC, respectively. The related *k-restricted Steiner tree problem* is that of finding a minimum cost Steiner tree whose full components (maximal subtrees whose leaves are all the terminal vertices and whose internal vertices are Steiner vertices) contain at most k terminal vertices.

Problem ST is an NP-hard problem [24, 26, 42]. As well, problem Steiner-2EC with edge costs all 1 (i.e., uniform edge costs) is NP-hard [51]. Thus, Steiner-kEC is an NP-hard problem.

For an early survey of Steiner tree heuristic results, see [64]. A very basic and rudimentary 2-approximation algorithm for ST simply finds the minimum spanning tree on the complete graph consisting of the terminal vertices T, with edge costs the cost of a minimum cost path between i and j in G, for all $i, j \in T$ [27, 49]. The approximation guarantee for ST was improved to $2(1 - \frac{1}{|T|})$ [59], then to 11/6 [65], then to 16/9 [7], and so on [41, 52, 53, 66], to 1.598 [35]. Robins and Zelikovsky [55] then obtained an approximation guarantee of $1 + \frac{\ln 3}{2} \approx 1.55$ for ST, and a 1.28-approximation guarantee for quasi-bipartite graphs (graphs in which no two Steiner vertices are adjacent) and for complete graphs with edge costs in $\{0, 1\}$. The long-standing best known upper bound on the integrality gap for the LP relaxation of ST was 2 [37]. Very recently, Bykra et al. broke through this stalemate, presenting a hypergraph-based LP relaxation for ST that has an integrality gap of 1.55 [8] (an alternate proof and analysis is provided by [11]). Not only did this improve the integrality gap for ST to less than 2 for general graphs, but also it matched the best known approximation guarantee of 1.55 for ST. Bykra et al. also presented an 1.39-approximation algorithm for ST, which is currently the best known. It currently remains an open problem to find an LP relaxation for ST that has an integrality gap of 1.39.

Generalization of SNDP

In a seminal work in 2001, Jain presented an important larger class of network design problems. Essentially, Jain's problem is that of finding, in a multi-graph that may not be simple and has non-negative edge costs, a minimum cost multi-subgraph having at least a specified number of edges in each cut [37]. In his work, Jain introduced an innovative and elegant technique (now known as iterated LP rounding), in order to obtain a 2-approximation algorithm for his problem. In addition to providing an approximation guarantee, Jain shows that his algorithm gives a constructive proof that the integrality gap of the linear programming relaxation of his general problem is at most 2 for all special cases of it. In many cases, these results provided improvements on a large number of well known edge connected network problems which are included as special cases of his problem [37]. As we refer to Jain's problem a number of times in this dissertation, we briefly describe it here.

The following definition is needed in order to state Jain's problem: A function $h : 2^{\mathcal{V}} \longrightarrow \mathbb{Z}$ is *weakly supermodular* if (1) $h(\mathcal{V}) = 0$, and (2) for every two sets $A, B \subseteq \mathcal{V}$, at least one of the following conditions holds: $h(A) + h(B) \leq h(A \backslash B) + h(B \backslash A)$, or $h(A) + h(B) \leq h(A \cap B) + h(B \cup A)$. The general network problem stated by Jain is as follows [37]: Given a multi-graph $G' = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, and a weakly supermodular cut requirement function $f : 2^V \longrightarrow \mathbb{Z}$, find a minimum cost spanning multi-subgraph of G' such that there are at least $f(S)$ edges in every cut $\delta(S)$, for all $S \subset V$. In other words, in a feasible solution to Jain's problem, there are at least $f(S)$ edge-disjoint paths between vertices u, v, where $u \in S$, $v \in V \backslash S$ [28]. Problem ST is a special case of Jain's general network problem – simply set $f(S) = 1$ if $S \subset V$ separates vertices of T (i.e., S and $V \backslash S$ both contain some vertices in T), and set $f(S) = 0$ otherwise. Problem SNDP is the special case where $f(S) := \max_{ij \in \delta(S)} r_{ij}$ for all $S \subset V$.

It is of interest that Jain provides an asymptotically tight example showing that the approximation guarantee of his general network problem is exactly 2. This example is as follows [37]: Consider the graph consisting of the wheel on the vertex set $V = \{c, a_1, a_2, \ldots, a_n\}$, where vertex c is the centre of the wheel and vertices $a_1, a_2, \ldots, a_n$ are on the "rim". The edges incident to c (the "spoke" edges) each have cost 1, and the "rim" each have cost $2 - \epsilon$. See Figure 1.3. The connectivity cut requirements are as follows:

$$\begin{aligned} x(\delta(S)) &\geq 1 \ \ \forall\, S \subset V;\ S \neq \{c\},\ V\backslash\{c\} \\ x(\delta(S)) &\geq 0 \ \ \forall\, S = \{c\},\ V\backslash\{c\} \end{aligned}$$

Notice that this example is in fact a special case of ST with metric edge costs.

As shown in Figure 1.3, the solution given by Jain's approximation algorithm choses a path of $n-1$ outside "rim" edges, and thus is of total cost $(n-1)(2-\epsilon) = 2n - \epsilon n - 2 + \epsilon \approx 2n - 2$; while an optimal solution consists of the n "spoke" edges, and has a total cost of n. The ratio of the cost of the heuristic solution over the cost of an optimal solution is then $\frac{2n-2}{n} = 2 - \frac{2}{n}$, which asymptotically approaches 2 as n gets large. Thus, for this example, Jain's approximation algorithm asymptotically reaches the approximation guarantee of 2. Furthermore, an optimal solution to the LP relaxation on this example consists of solution values of 1/2 for all the "rim"

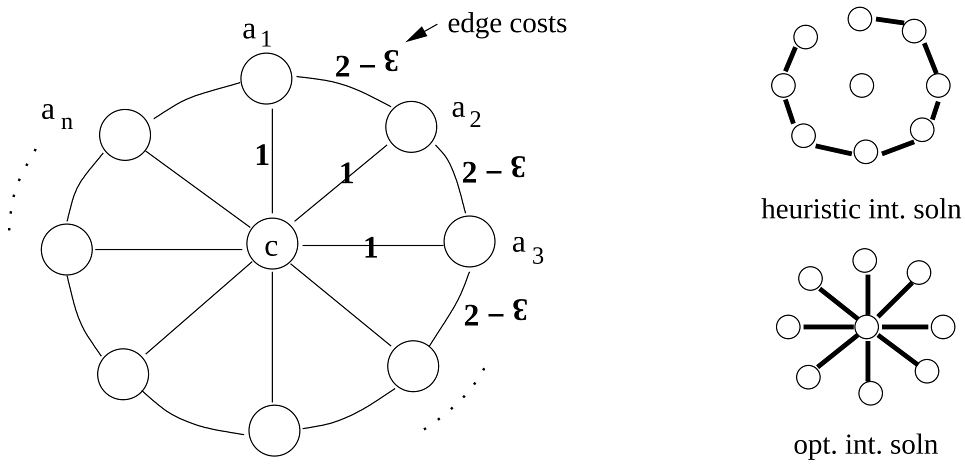

Figure 1.3: An example on which Jain's approximation algorithm asymptotically reaches an approximation guarantee of 2 [37].

edges, and 0 for all the "spoke" edges. The cost of this is $\frac{1}{2}(2-\epsilon)n = (1-\frac{\epsilon}{2})n$. Thus, the integrality ratio of the LP relaxation of this wheel example is $\frac{n}{(1-\frac{\epsilon}{2})n} = \frac{1}{1-\frac{\epsilon}{2}}$, which asymptotically approaches 1 as ϵ approaches 0. Thus, the integrality ratio of the LP relaxation of Jain's wheel example asymptotically approaches 1.

New: Vital Core Connectivity Problem

In this dissertation we introduce a new network design problem, which we call the *Vital Core Connectivity Problem* (VCC). 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 $k, l \in \mathbb{Z}$, $1 \le l \le k$, 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 (Chapter 2, Section 2), VCC is, in general, an NP-hard problem and is a generalization of MST and kEC. 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). It is the problem of finding a minimum cost spanning tree of G that 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 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 graph disconnects that graph). When $l = 1$ and the secondary vertices $V \backslash V^*$ consist 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 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 (denoted multi-kEC).

The VCC is related to, but distinct from, other connectivity problems. A comparison is made in Chapter 2, the chapter pertaining to VCC. We will also 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$(1, 1, V^*)$.

Bibliography

[1] M. Aggarwal and N. Garg, A Scaling Technique for Better Network Design, Proc. of the 5th annual ACM-SIAM Symposium on Discrete Algorithms (1994) 233-240.

[2] A. Agrawal, P. Klein, R. Ravil, When Trees Collide: An Approximation Algorithm for the Generalized Steiner Problem on Networks, SIAM J. COMPUT. 24 No.3 (1995) 440-456. First appeared in ACM Poceedings (1991) 134-144.

[3] R. Albert, A.-L. Barabási, Statistical Mechanics of Complex Networks, Review of Modern Physics 74 (2002) 47-96.

[4] A. Alexander, S. Boyd, P. Elliott-Magwood, On the Integrality Gap of the 2-Edge Connected Subgraph Problem, Technical Report TR-2006-04, S.I.T.E., University of Ottawa, Ottawa (2006).

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

[6] 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.

[7] P. Berman, V. Ramaiyer, Improved Approximations for the Steiner Tree Problem, Proc. 3rd ACM-SIAM Symposium on Discrete Algorithms (1992) 325-334.

[8] J. Byrka, F. Grandoni, T. Rothvoβ, L. Sanità, An Improved LP-based Approximation for Steiner Tree, STOC 2010, Cambridge, Massachusetts, USA (2010).

[9] R. Carr, R. Ravi, A New Bound for the 2-Edge Connected Subgraph Problem, Lecture Notes in Computer Science (IPCO Conference VI) 1412 (1998) 112-125.

[10] D. Chakrabarty, N. Devanur, V. Vazirani, New Geometry-inspired Relaxations and Algorithms for the Metric Steiner Tree Problem, Math Programming Series A (2009). Preliminary version appeared in the Proceedings of the 13th Conference on Integer Programming and Combinatorial Optimization (IPCO) (2008).

[11] D. Chakrabarty, J. Könemann, D. Pritchard, Integrality Gap of the Hypergraphic Relaxation of Steiner Trees: A Short Proof of a 1.55 Upper Bound, Operations Research Letters 38 (2010) 567-570.

[12] J. Cheriyan, A. Sebo, Z. Szigeti, Improving on the 1.5-Approximation of a Smallest 2-Edge Connected Spanning Subgraph, SIAM Journal of Discrete Mathematics 14 (2001) 170-180.

[13] S. Chopra, M.R. Rao, The Steiner Tree Problem I: Formulations, Compositions and Extension of Facets, Mathematical Programming 64 (1994) 209-229.

[14] The Clay Mathematics Institute, P vs NP problem, http://www.claymath.org/millennium/P_vs_NP/. Accessed January 2012.

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

[16] P. Crescenzi, V. Kann, A Compendium of NP Optimization Problems, in: G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, chetti Spaccamela, A. Mar, M. Protasi, Approximate Solution of NP-hard Optimization Problems, Springer-Verlag (1998). See http://www.nada.kth.se/theory/compendium/

[17] Y. Dinitz, Z. Nutov, A 3-Approximation Algorithm for Finding Optimum 4,5-Vertex-Connected Spanning Subgraphs, Journal of Algorithms 32 (1999) 31-40.

[18] D. Eppstein, ICS 161: Design and Analysis of Algorithms Lecture notes, Dept Information & Computer Science, University of California, http://www.ics.uci.edu/~eppstein/161/960312.html. Accessed March 2011.

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

[20] L. Fleischer, K. Jain, D. Williamson, Iterative Rounding 2-approximation Algorithms for Minimum-cost Vertex Connectivity Problems, Journal of Computer and Systems Sciences 72 (2006) 838-867.

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

[22] H.N. Gabow, An Improved Analysis for Approximation the Smallest k-edge Connected Spanning Subgraph of a Multi-graph, SIAM J. Discrete Mathematics 19 No.1 (2005) 1-18.

[23] H.N. Gabow, M.X. Goemans, E. Tardos, D.P. Williamson, Approximating the smallest k-edge connected spanning subgraph by LP-rounding, Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms (2005) 562-571.

[24] M.R. Garey, R.L. Graham, D.S. Johnson, The Complexity of Computing Steiner Minimal Trees, SIAM J. Appl. Math. 32 (1977) 835-859.

[25] H.N. Gabow, M.X. Goemans, D.P. Williamson, An efficient approximation algorithm for the survivable network design problem, Mathematical Programming 82 (1998) 13-40.

[26] 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).

[27] E.N. Gilbert and H.O. Pollak, Steiner Minimal Trees, SIAM J. Appl. Math. 16 (1968) 1-29.

[28] M.X. Goemans and D.J. Bertsimas, Survivable networks, linear programming relaxations and the parsimonious property, Mathematical Programming 60 (1993) 145-166.

[29] M. Goemans, Y. Myung, A Catalog of Steiner Tree Formulations, Networks 23 (1993) 19-28.

[30] T.F. Gonzalez, ed., Handbook of Approximation Algorithms and Metaheuristics, Chapman & Hall/CRC (2007).

[31] M. Grötschel, C.L. Monma, M. Stoer, Design of Survivable Networks, in: M.O. Bell, T.L. Magnant, C.L. Monma, G.L. Nemhauser, eds, Network Models (Handbooks in OR & MS, Vol.7), North-Holland (1995).

[32] M. Grötschel, C.L. Monma, M. Stoer, Polyhedral Approaches in Network Survivability, DIMACS Series in Discrete Mathematics and Theoretical Computer Science 5 (1991) 121-141.

[33] P. Gubbala, B. Raghavachari, A 4/3-Approximation Algorithm for Minimum 3-Edge-Connectivity, WADS 2007, LNCS 4619 (2007) 39-51.

[34] F. Harary, The Maximum Connectivity of a Graph, Proc. of the National Academy of Sciences of the USA 48 No.7 (1962) 1142-1146.

[35] S. Hougardy, H. J. Promel, A 1.598 Approximation Algorithm for the Steiner Problem in Graphs, Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, Philadelphia, ACM, New York (1999) 448-453.

[36] T. Hsu, M. Kao, A Unifying Augmentation Algorithm for Two-Edge Connectivity and Biconnectivity, Journal of Combinatorial Optimization 2 No.3 (1998) 237-256.

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

[38] 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) 484-489.

[39] H. Kang, C. Bulter, J. Chen, A New Survivability Measure for Military Communication Networks, 1998 IEEE (1998) 71-76.

[40] R.M. Karp, Reducibility Among Combinatorial Problems, in: R.E. Miller, J.W. Thatcher, eds, Complexity of computer computations, Plenum Press (NY) (1972) 85-104.

[41] M. Karpinski, A. Zelikovsky, New Approximation Algorithms for the Steiner Tree Problem, Journal of Combinatorial Optimization 1 (1997) 47-65.

[42] S. Khuller, Approximation Algorithms for Finding Highly Connected Subgraphs, in: Approximation Algorithms for NP-hard Problems, PWS Publishing Company, Boston (1997) 236-265.

[43] S. Khuller and U. Vishkin, Biconnectivity Approximations and Graph Carvings, Journal of the ACM 41 (1994) 214-235. Preliminary version in Proc. 24th Annual ACM Symp. on Theory of Computing (1992) 759-770.

[44] J. Könemann, D. Pritchard, K. Tan, A Partition-based Relaxation for Steiner Trees, Math. Programming Series A (2009).

[45] G. Kortsarz, R. Krauthgamer, J.R. Lee, Hardness of Approximation for Vertex-Connectivity Network-Design Problems, APPROX 2002, LNCS 2462 (2002) 185-199.

[46] G. Kortsarz, Z. Nutov, Approximating Minimum-Cost Connectivity Problems, in: T.F. Gonzalez, ed., Handbook of Approximation Algorithms and Metaheuristics, Chapman & Hall (2007).

[47] G. Kortsarz, Z. Nutov, Approximating Node Connectivity Problems via Set Covers, Algorithmica 37 (2003) 75-92.

[48] G. Kortsarz, Z. Nutov, Approximation Algorithm for k-node Connected Subgraphs via Critical Graphs, extended abstract, STOC04 (2004) 138-145.

[49] L. Kou, G. Markowsky, L. Berman, A fast Algorithm for Steiner trees, Acta Inform. 15 (1981) 141-145.

[50] P. Krysta, Approximating Minimum Size $\{1, 2\}$-connected Networks, Discrete Applied Mathematics 125 (2003) 267-288. First appeared in: P. Krysta, A. Kumar, Approximation Algorithms for Minimum Size 2-Connectivity Problems, STACS (2001) 431-442.

[51] C.L. Monma, B.S. Munson, W.R. Pulleyblank, Minimum-weight Two-connected Spanning Networks, Mathematical Programming 46 (1990) 153-171.

[52] H.J. Prömel, A. Steger, A New Approximation Algorithm for the Steiner Tree Problem with Performance Ratio 5/3, Journal of Algorithms 36 (2000) 89-101.

[53] H.J. Prömel, A. Steger, RNC-approximation Algorithms for the Steiner Problem, Proceedings of the 14th Annual Symposium on Theoretical Aspects of Computer Science, Springer, Berlin (1997) 559-570.

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

[55] G. Robins, A. Zelikovsky, Tighter Bounds for Graph Steiner Tree Approximation, SIAM J. Discrete Math 19 No.1 (2005) 122-134.

[56] A. Schrijver, On the History of Combinatorial Optimization (till 1960), in: "Handbook of Discrete Optimization", K. Aardal, G.L. Nemhauser, R. Weismantel, eds., Elsevier, Amsterdam (2005) 1-68.

[57] S. Soni, R. Gupta, H. Pirkul, Survivable Network Design: The State of the Art, Information Systems Frontiers 1:3, (1999) 303-315.

[58] K. Sunitha, N. Kalyani, Formal Languages & Automata Theory, Tata McGraw-Hill Education (2010).

[59] H. Takahashi and A. Matsuyama, An Approximate Solution for the Steiner Problem in Graphs, Math, Japonica 24 (1980) 573-577.

[60] V.V. Vazirani, Recent Results on Approximating the Steiner Tree Problem and its Generalizations, Theoretical Computer Science 235 (2000) 205-216.

[61] V. Vazirani, Approximation Algorithms, Springer-Verlag Berlin Heidelberg New York (2003).

[62] S. Vempala, A. Vetta, Factor $\frac{4}{3}$ Approximations for Minimum 2-Connected Subgraphs, Proceedings of the 3rd International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, Max-Planck-Institut f"ur Informatik, Saarbŕucken, Germany, Springer, Berlin, (2000) 262-273.

[63] David Williamson, David Shmoys, The Design of Approximation Algorithms, Cambridge University Press (2011).

[64] P. Winter, Steiner Problem in Networks: A Survey, Networks 17 (1987) 129-167.

[65] A. Z. Zelikovsky, The 11/6 Approximation Algorithm for the Steiner Problem on Networks, Algorithmica 9 (1993) 463-470.

[66] A. Zelikovsky, Better Approximation Bounds for the Network and Euclidean Steiner Tree Problems, Tech. report CS-96-06, University of Virginia, Charlottesville, VA (1996).

Chapter 2

The Vital Core Connectivity Problem

• Sylvia Boyd and Amy Cameron. The Vital Core Connectivity Problem, Technical Report, University of Ottawa, 2011.

Work of Amy Cameron under the supervision of Sylvia Boyd. Results are joint work of Amy Cameron and Sylvia Boyd.

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.

2.1 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

[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].

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 is a feasible solution to VCC(k, l, V^*). However, as shown in Figure 2.1, such a solution, while being a feasible solution to VCC(k, l, V^*), is not necessarily an optimal solution to VCC(k, l, V^*). In Figure 2.1(a), we show a complete weighted graph on 5 vertices. In Figure 2.1(b) and 2.1(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

[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.

multi-AUG solutions shown in Figure 2.1(b) and (c) are both feasible solutions to VCC$(2, 1, V^*)$, but only Figure 2.1(c) is an optimal solution to 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 VCC(k, l, V^*).

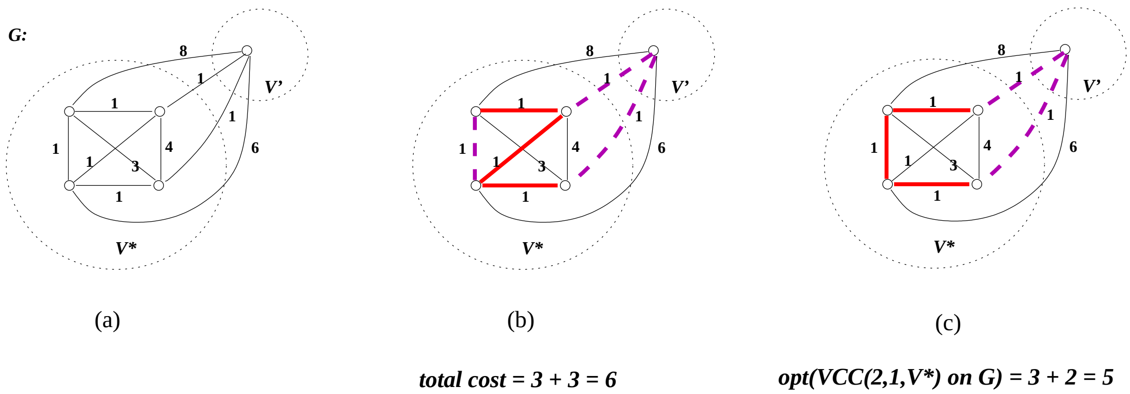

Figure 2.1: Example showing that finding a minimum cost multi-l-edge connected spanning subgraph on $G[V^*]$, and then augmenting it at minimum cost to a multi-k-edge connected spanning subgraph on G, does not always yield an optimal solution to VCC(k, l, V^*).

An outline of this paper is as follows: In Section 2.2 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 EMST. We also give a complete linear description for EMST. In Section 2.3.1, we present the results necessary for our approximation algorithm and integrality gap results for the general VCC(k, l, V^*). In Section 2.3.2, we present Algorithm VCC, an approximation algorithm for VCC(k, l, V^*); and present an upper bound on the problem's LP relaxation. In Section 2.3.3, 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 2.2.1 and 2.2.2 are fundamental to Section 2.3. We conclude the section by making some final observations and comparisons. In the final section, Section 2.4, 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 $opt_G(\mathcal{Q})$ 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].

2.2 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 2.2. Let $V^* = V \backslash V' (= \hat{V})$. Let $L := Q$.

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

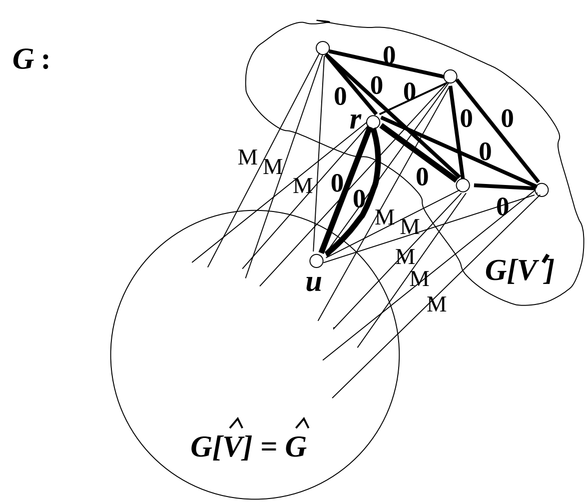

Figure 2.2: Illustration of G in the proof of Theorem 2. Edges in $\delta_G(V')\backslash\{ru\}$ have cost M.

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^*). ∎

2.2.1 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 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 2.3). 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).

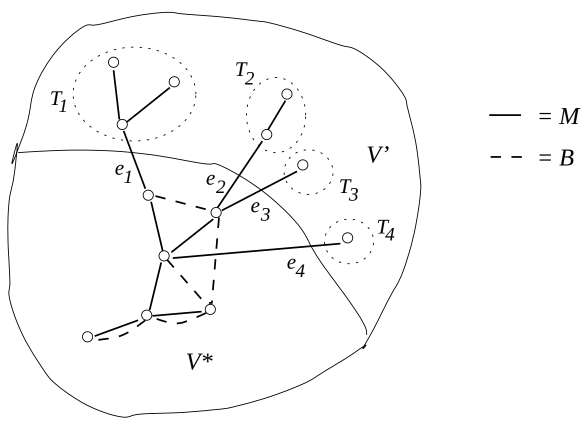

Figure 2.3: Replacing $M[V^*]$ by B in M.

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 $O(|V|^2)$. ■

2.2.2 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* $P = (V_1, V_2, \ldots, V_{k_P})$ 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 $i, j \in \{1, 2, \ldots, k_P\}$, $i \neq j$. Notice that in such a partition, each *part* V_i induces a complete subgraph $G[V_i]$ for all $i = 1, 2, \ldots, k_P$. 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, $i = 1, 2, \ldots, k_P$, 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.

[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.

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.

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*

$$\begin{aligned} & \textit{minimize} \;\; cx && && \qquad (2.1)\\ & \textit{subject to} \;\; \sum_{e \in E_P} x_e && \geq k_P - 1, && \textit{for all partitions } P \textit{ of } G, \qquad (2.2)\\ & x_e && \geq 0, && \textit{for all } e \in E. \qquad (2.3) \end{aligned}$$

Moreover, the feasible region of LP(2.1) 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^*):

$$\begin{aligned} & \text{minimize} \;\; cx && && \qquad (2.4)\\ & \text{subject to} \;\; \sum_{e \in E_P} x_e && \geq k_P - 1, && \text{for all partitions } P \text{ of } G/V^*, \qquad (2.5)\\ & \sum_{e \in E_P} x_e && \geq k_P - 1, && \text{for all partitions } P \text{ of } G[V^*], \qquad (2.6)\\ & x_e && \geq 0, && \text{for all } e \in E. \qquad (2.7) \end{aligned}$$

Notice that constraints (2.5) and (2.7) are precisely those constraints for the minimum spanning tree LP(2.1) on the shrunk graph G/V^* of G, and constraints (2.6) and (2.7) are precisely those constraints for the minimum spanning tree LP(2.1) on the vital core $G[V^*]$.

We will now show that LP(2.4) 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(2.1).

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(2.4).*

Proof: We show that every feasible solution to the constraints (2.5)-(2.7) 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 (2.5)-(2.7), i.e., x is a point in the feasible region determined by the constraints (2.5)-(2.7). 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

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

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 (2.5)-(2.7) 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(2.4) that is the incidence vector of a minimum cost spanning tree of G. Thus, LP(2.4) 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(2.4).*

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 (2.4) that is the incidence vector of a minimum cost spanning tree of G. ■

2.3 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:

$$\text{minimize} \quad cx \tag{2.8}$$

$$\text{subject to} \quad x(\delta_G(S)) \geq k, \quad \text{for all } \emptyset \subset S \subset V, \tag{2.9}$$

$$x(\delta_{G[V^*]}(S)) \geq l, \quad \text{for all } \emptyset \subset S \subset V^*, \tag{2.10}$$

$$x_e \geq 0, \quad \text{for all } e \in E, \tag{2.11}$$

$$x_e \quad \text{integer, for all } e \in E. \tag{2.12}$$

The constraints (2.9) ensure that the multi-subgraph is k-edge connected; and the constraints (2.10) 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(2.8) without the integer constraints (2.12), 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(2.8) without the constraints (2.10) and (2.12).

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 (2.10) 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 (2.9) and (2.11), which is precisely the LP of multi-$k\text{EC}_{LP}$. 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 $V^* = V$.

2.3.1 Preliminary Results

Before proceeding to our approximation algorithm and an upper bound on the integrality gap of the LP relaxation of $\text{VCC}(k, l, V^*)$, 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*

$$\begin{aligned} \textit{minimize} \quad & cx && (2.13)\\ \textit{subject to} \quad x(\delta(S)) \;\geq\; & 1, \quad \textit{for all } S \subseteq V \textit{ s.t.} \\ & \quad |S \cap T| \textit{ odd}, && (2.14)\\ x_e \;\geq\; & 0, \quad \textit{for all } e \in E. && (2.15) \end{aligned}$$

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

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

Proof: Let x^* be an optimal solution to LP(VCC). In particular, x^* satisfies the constraints (2.9). 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 (2.14). Additionally, since x^* satisfies the constraints (2.11), $\frac{1}{k}x^*$

also satisfies the constraints (2.15). Therefore, $\frac{1}{k}x^*$ satisfies the constraints (2.14) and (2.15), and thus is a feasible solution of LP(2.13). Using Theorem 11, we have that

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

$\blacksquare$

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

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

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

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

Lemma 14 *The following holds on G:*

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

Proof: Let x^* be an optimal solution to LP(VCC) on G and let $x' = \frac{2}{l}x^*$. Since x^* satisfies the constraints (2.9) and (2.10), and since $k \geq l$, therefore x' satisfies

$$x'(\delta(S)) \ \geq \ \frac{2k}{l} \ \geq \ 2, \qquad \text{for all } \emptyset \subset S \subset V \tag{2.16}$$

and

$$x'(\delta_{G[V^*]}(S)) \ \geq \ \frac{2l}{l} \ = \ 2, \qquad \text{for all } \emptyset \subset S \subset V^*. \tag{2.17}$$

Let $P = (S_1, S_2, \ldots, S_{k_P})$ be a partition of G. For x' on G we have, using (2.16):

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

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 (2.5).

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 (2.17):

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

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 (2.6).

Clearly, x' satisfies the constraints (2.7). Thus, $x' = \frac{2}{l}x^*$ is a feasible solution of LP(2.4), 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:*

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

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. ■

2.3.2 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 2.3.3).

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 2.2.1, 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:

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

$$\begin{aligned}
cost(H) &= \left\lceil \frac{k}{2} \right\rceil \cdot opt_G(EMST(V^*)) + \left(l - \left\lceil \frac{k}{2} \right\rceil\right) \cdot opt_{G[V^*]}(\tilde{T}\text{-}join) \\
&\quad + \left\lfloor \frac{k}{2} \right\rfloor \cdot opt_G(\hat{T}\text{-}join) \\
&\leq \left(\left\lceil \frac{k}{2} \right\rceil \cdot \frac{2}{l} + \left(l - \left\lceil \frac{k}{2} \right\rceil\right) \cdot \frac{1}{l} + \left\lfloor \frac{k}{2} \right\rfloor \cdot \frac{1}{k} \right) \\
&\quad \cdot opt_G(VCC(k, l, V^*)_{LP}). \qquad (2.18)
\end{aligned}$$

Case 1a: If k is even, (2.18) becomes

$$\begin{aligned}
cost(H) &\leq \left(\frac{k}{2} \cdot \frac{2}{l} + \left(l - \frac{k}{2}\right) \cdot \frac{1}{l} + \frac{k}{2} \cdot \frac{1}{k} \right) \cdot opt_G(VCC(k, l, V^*)_{LP}) \\
&= \left(\frac{k}{l} + 1 - \frac{k}{2l} + \frac{1}{2} \right) \cdot opt_G(VCC(k, l, V^*)_{LP}) \\
&= \frac{1}{2}\left(3 + \frac{k}{l}\right) \cdot opt_G(VCC(k, l, V^*)_{LP}). \qquad (2.19)
\end{aligned}$$

Case 1b: If k is odd, (2.18) becomes

$$\begin{aligned}
cost(H) &\leq \left(\left(\frac{k+1}{2}\right) \cdot \frac{2}{l} + \left(l - \frac{k+1}{2}\right) \cdot \frac{1}{l} + \left(\frac{k-1}{2}\right) \cdot \frac{1}{k} \right) \cdot opt_G(VCC(k, l, V^*)_{LP}) \\
&= \left(\frac{2k}{2l} + \frac{2}{2l} + 1 - \frac{k}{2l} - \frac{1}{2l} + \frac{1}{2} - \frac{1}{2k} \right) \cdot opt_G(VCC(k, l, V^*)_{LP}) \\
&= \frac{1}{2}\left(\left(3 + \frac{k}{l}\right) + \left(\frac{1}{l} - \frac{1}{k}\right) \right) \cdot opt_G(VCC(k, l, V^*)_{LP}). \qquad (2.20)
\end{aligned}$$

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

$$\begin{aligned} cost(H) &= \left\lceil \frac{l}{2} \right\rceil \cdot opt_G(EMST(V^*)) + \left\lfloor \frac{l}{2} \right\rfloor \cdot opt_{G[V^*]}(\tilde{T}\text{-}join) + \left\lfloor \frac{l}{2} \right\rfloor \cdot opt_G(\hat{T}\text{-}join) \\ &\quad + \left\lfloor \frac{k-l}{2} \right\rfloor \cdot opt_G(T\text{-}join) + \left\lceil \frac{k-l}{2} \right\rceil \cdot opt_G(MST) \\ &\leq \left(\left\lceil \frac{l}{2} \right\rceil \cdot \frac{2}{l} + \left\lfloor \frac{l}{2} \right\rfloor \cdot \frac{1}{l} + \left(\left\lfloor \frac{l}{2} \right\rfloor + \left\lfloor \frac{k-l}{2} \right\rfloor \right) \cdot \frac{1}{k} + \left\lceil \frac{k-l}{2} \right\rceil \cdot \frac{2}{k} \right) \\ &\quad \cdot opt_G(VCC(k,l,V^*)_{LP}). \end{aligned} \tag{2.21}$$

Case 2a: When k, l are even, (2.21) becomes

$$\begin{aligned} cost(H) &\leq \left(\frac{l}{2} \cdot \frac{2}{l} + \frac{l}{2} \cdot \frac{1}{l} + \left(\frac{l}{2} + \frac{k-l}{2} \right) \cdot \frac{1}{k} + \left(\frac{k-l}{2} \right) \cdot \frac{2}{k} \right) \\ &\quad \cdot opt_G(VCC(k,l,V^*)_{LP}) \\ &= \left(1 + \frac{1}{2} + \frac{l}{2k} + \frac{1}{2} - \frac{l}{2k} + 1 - \frac{l}{k} \right) \cdot opt_G(VCC(k,l,V^*)_{LP}) \\ &= \left(3 - \frac{l}{k} \right) \cdot opt_G(VCC(k,l,V^*)_{LP}). \end{aligned} \tag{2.22}$$

Case 2b: When l is odd and k is even, (2.21) becomes

$$\begin{aligned} cost(H) &\leq \left(\left(\frac{l+1}{2} \right) \cdot \frac{2}{l} + \left(\frac{l-1}{2} \right) \cdot \frac{1}{l} + \left(\frac{l-1}{2} + \frac{k-l-1}{2} \right) \cdot \frac{1}{k} \right. \\ &\quad \left. + \left(\frac{k-l+1}{2} \right) \cdot \frac{2}{k} \right) \cdot opt_G(VCC(k,l,V^*)_{LP}) \\ &= \left(1 + \frac{1}{l} + \frac{1}{2} - \frac{l}{2l} + \frac{l}{2k} - \frac{1}{2k} + \frac{1}{2} - \frac{l}{2k} - \frac{1}{2k} + 1 - \frac{l}{k} + \frac{1}{k} \right) \\ &\quad \cdot opt_G(VCC(k,l,V^*)_{LP}) \\ &= \left(\left(3 - \frac{l}{k} \right) + \frac{1}{2l} \right) \cdot opt_G(VCC(k,l,V^*)_{LP}). \end{aligned} \tag{2.23}$$

Case 2c: When l is even and k is odd, (2.21) becomes

$$\begin{aligned} cost(H) &\leq \left(\frac{l}{2} \cdot \frac{2}{l} + \frac{l}{2} \cdot \frac{1}{l} + \left(\frac{l}{2} + \frac{k-l-1}{2} \right) \cdot \frac{1}{k} \right. \\ &\qquad \left. + \left(\frac{k-l+1}{2} \right) \cdot \frac{2}{k} \right) \cdot opt_G(\mathit{VCC}(k,l,V^*)_{LP}) \\ &= \left(\left(3 - \frac{l}{k} \right) + \frac{1}{2k} \right) \cdot opt_G(\mathit{VCC}(k,l,V^*)_{LP}). \end{aligned} \tag{2.24}$$

Case 2d: When k, l are odd, (2.21) becomes

$$\begin{aligned} cost(H) &\leq \left(\left(\frac{l+1}{2} \right) \cdot \frac{2}{l} + \left(\frac{l-1}{2} \right) \cdot \frac{1}{l} + \left(\frac{l-1}{2} + \frac{k-l}{2} \right) \cdot \frac{1}{k} \right. \\ &\qquad \left. + \left(\frac{k-l}{2} \right) \cdot \frac{2}{k} \right) \cdot opt_G(\mathit{VCC}(k,l,V^*)_{LP}) \\ &= \left(\left(3 - \frac{l}{k} \right) + \frac{1}{2l} - \frac{1}{2k} \right) \cdot opt_G(\mathit{VCC}(k,l,V^*)_{LP}). \end{aligned} \tag{2.25}$$

Therefore,

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

i.e.,

$$\frac{opt_G(\mathit{VCC}(k,l,V^*))}{opt_G(\mathit{VCC}(k,l,V^*)_{LP})} \leq \omega_{k,l}, \tag{2.26}$$

where $\omega_{k,l}$ is the bound from equation (2.19), (2.20), (2.22), (2.23), (2.24), or (2.25), 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 $\mathit{VCC}(k,l,V^*)$ that is returned by Algorithm VCC on G. Then, on G,*

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

Proof: The first inequality follows directly from equations (2.19), (2.20), (2.22), and (2.23). 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. ■

2.3.3 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.

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 2.2.1, 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,

$$\begin{aligned} cost(H) &= 1 \cdot opt_G(EMST(V^*)) + 1 \cdot opt_G(\hat{T}\text{-}join) \\ &\quad + \left\lfloor \frac{k-2}{2} \right\rfloor \cdot opt_G(T\text{-}join) + \left\lceil \frac{k-2}{2} \right\rceil \cdot opt_G(MST) \\ &\leq \left(1 \cdot 1 + \left(1 + \left\lfloor \frac{k-2}{2} \right\rfloor \right) \cdot \frac{1}{k} + \left\lceil \frac{k-2}{2} \right\rceil \cdot \frac{2}{k} \right) \\ &\quad \cdot opt_G(VCC(k, 1, V^*)_{LP}). \qquad (2.27) \end{aligned}$$

When k is even, (2.27) becomes

$$\begin{aligned} cost(H) &\leq \left(1 + \left(1 + \frac{k-2}{2} \right) \cdot \frac{1}{k} + \left(\frac{k-2}{2} \right) \cdot \frac{2}{k} \right) \cdot opt_G(VCC(k, 1, V^*)_{LP}) \\ &= \left(\frac{5}{2} - \frac{2}{k} \right) \cdot opt_G(VCC(k, 1, V^*)_{LP}). \qquad (2.28) \end{aligned}$$

When k is odd, (2.27) becomes

$$\begin{aligned} cost(H) &\leq \left(1 + \left(1 + \frac{k-2-1}{2} \right) \cdot \frac{1}{k} + \left(\frac{k-2+1}{2} \right) \cdot \frac{2}{k} \right) \cdot opt_G(VCC(k, 1, V^*)_{LP}) \\ &= \left(\frac{5}{2} - \frac{3}{2k} \right) \cdot opt_G(VCC(k, 1, V^*)_{LP}). \qquad (2.29) \end{aligned}$$

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 (2.28), and (2.29). 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,

$$opt_{G[V^*]}(MST) \ \leq \ opt_G(EMST(V^*)) \ \leq \ \frac{2}{l} \, opt_G(VCC(k, l, V^*)). \tag{2.30}$$

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 $G[V^*]$,

$$opt_{G[V^*]}(\text{multi-}l\text{EC}) \ \leq \ opt_G(VCC(k, l, V^*)). \tag{2.31}$$

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

$$opt_G(\text{multi-}k\text{EC}) \ \leq \ opt_G(VCC(k, l, V^*)). \tag{2.32}$$

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 (2.30) 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 (2.31) and (2.32), 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^*).

2.4 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 2.2.

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 $\mathcal{V} = \{V_1^*, V_2^*, \ldots, V_q^*\}$ be a set of non-empty pairwise disjoint subsets of vertices such that $V_m^* \subseteq V$ for $m = 1, 2, \ldots, q$. Let 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 $m = 1, 2, \ldots, q$.*

Proof: Consider the subset of vital vertices V_m^*, $m \in \{1, 2, \ldots, q\}$. 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 $m \in \{1, 2, \ldots, q\}$, 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 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 2.4. It is easy to see that $\hat{M}$ is a spanning tree of G. Furthermore, $\hat{M}[V_m^*]$ is connected, since $\hat{M}[V_m^*]$ 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.

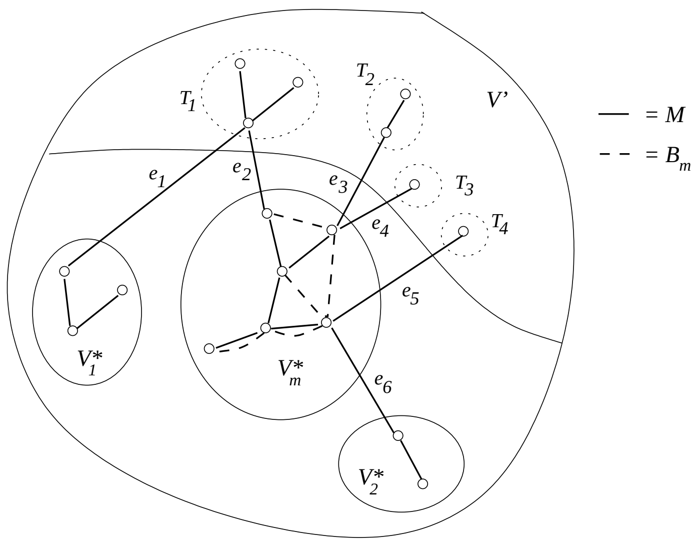

Figure 2.4: Replacing $M[V^*]$ by B_m in M.

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 2.4.

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 $O(|V|^2)$. ■

[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.

A complete linear description of EMST is given in Section 2.2.2. Using the same techniques as above, this can be extended naturally to form a complete linear description of VCMST:

$$
\begin{aligned}
\text{minimize} \quad & cx && (2.33)\\
\text{subject to} \quad & \sum_{e \in E_P} x_e \geq k_P - 1, \ \text{for all partitions } P \text{ of } G/\mathcal{V}, && (2.34)\\
& \sum_{e \in E_P} x_e \geq k_P - 1, \ \text{for all partitions } P \text{ of} && (2.35)\\
& \qquad G[V_1^*], G[V_2^*], \ldots, G[V_q^*], \ \text{respectively,} &&\\
& x_e \geq 0, \quad \text{for all } e \in E. && (2.36)
\end{aligned}
$$

Constraints (2.34) and (2.36) are precisely those constraints for the minimum spanning tree LP(2.1) on the shrunk graph $G/\mathcal{V}$ of G, and constraints (2.35) and (2.36) are precisely those constraints for the minimum spanning tree LP(2.1) on the vital cores $G[V_1^*], G[V_2^*], \ldots, G[V_q^*]$, respectively.

2.5 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).

Bibliography

[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 2.3.3, 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
		...		

Chapter 3

Integrality Gap and Approximation Algorithm for the Multi-survivable Network Design Problem

• Sylvia Boyd and Amy Cameron. Integrality Gap and Approximation Algorithm for the Multi-survivable Network Design Problem, Technical Report TR-2009-03, University of Ottawa, 2009.

Work of Amy Cameron under the supervision of Sylvia Boyd. All results are joint work of Amy Cameron and Sylvia Boyd.

Abstract

Given a complete graph with non-negative edge costs and non-negative integer vertex connectivity requirements, we consider the problem of finding a minimum cost subgraph that has at least r_{ij} edge-disjoint paths between every pair of distinct vertices i, j, where r_{ij} is the minimum of the vertex connectivity requirements r_i and r_j. This is known as the survivable network problem. When multi-edges are allowed in the solution, we call this the multi-survivable network design problem. Let r_m be the maximum vertex connectivity requirement and r_l be the smallest nonzero vertex connectivity requirement. We show that the integrality gap for the linear programming relaxation of the multi-survivable network design problem is at most $\frac{3}{2}\frac{r_m}{r_l}$ when r_m is even, and at most $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ when r_m is odd. We also present an approximation algorithm with this bound as a performance guarantee. We show that these results are improvements for the best known bounds for certain special cases of this problem.

3.1 Introduction

The minimum cost k-edge connected subgraph problem, $k \geq 2$, is an important problem in network design. Given a network of centres and specific non-negative costs for connecting any two centres with a link, this problem consists of finding a cheapest way to construct a network so that it can survive the loss of $k-1$ links, i.e., so that the network remains connected even if any $k-1$ of the links are lost. The problem has many practical, cost-saving applications, such as the design of reliable communication and transportation networks. Here, "cost" can be interpreted as the construction cost, and "reliable" as the continuation of services in the event of a catastrophe causing the loss of a "link" (i.e. a communications cable or a transportation road). For such applications, finding a solution that is either optimal or very close to optimal means substantial financial savings for the company constructing the network.

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. For example, to connect two components of a network, it may be more practical and more cost efficient to use multiple duplicate links between two centres of the network components (one centre from each component) rather than a single link between two centres and other single links between different pairs of centres of the two network components. 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. Also, in case of a wire failure, sensors in nuclear power plants are each connected by three wires to the main system.

A more general problem in network design is the survivable network design problem. This problem consists of finding a cheapest way to construct a network so that it can survive the loss of certain numbers of links in certain areas of the network. This is used, for example, in designing fibre optic telecommunications networks [9, 12].

3.1.1 Multi-SNDP, SNDP, Metric kEC, and Multi-kEC Problems

Given a graph $G = (V, E)$, and any vertex set $S \subset V$, let $\delta(S)$ denote the set of edges $\{uv \in E : u \in S, v \notin S\}$.

The *(edge connected) survivable network design problem* (SNDP) (also called the *generalized Steiner network problem*) is as follows: Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. For each pair of distinct vertices $i, j \in V$, let the edge connectivity requirement be $r_{ij} = \min\{r_i, r_j\}$. The problem is to find a minimum-cost subgraph that has, for each pair of distinct vertices $i, j \in V$, at least r_{ij} edge-disjoint paths between i and j. In such a solution, the loss of any k edges still allows all vertices with connectivity requirement greater than k to be mutually connected.

If multiple copies of the edges are allowed in the solution of SNDP, the problem will be referred to as the *multi-survivable network design problem* (multi-SNDP). A *multi-subgraph* is a subgraph with multiple copies of at least one of its edges. The equivalent cut requirement version of multi-SNDP is: Given a cut requirement function $f : 2^V \to \mathbb{Z}$ defined on the subsets of V by $f(S) = \max_{i \in S,\ j \notin S} r_{ij}$ for all $S \subset V$, find a minimum-cost multi-subgraph having at least $f(S)$ edges in each cut $\delta(S)$, $S \subset V$. Note that if G is not complete, we can make it into a complete graph by adding in all the "missing" edges and assigning each of them an appropriately large cost.

When the edge costs $c \in \mathbb{R}^E$ of a graph are non-negative and satisfy the triangle inequality $c_{uw} \leq c_{uv} + c_{vw}$ for all distinct vertices $u, v, w \in V$, the costs are called *metric*. A SNDP problem with metric edge costs will be referred to as *metric SNDP*.

A connected graph is said to be *k-edge connected*, $k \geq 1$, $k \in \mathbb{Z}$, if any $k - 1$ edges can be removed without disconnecting the graph, or, equivalently, if there are at least k edge-disjoint paths between every pair of vertices, or, equivalently, if every cut $\delta(S), S \subset V$, has at least k edges.

Given a complete graph, $G = (V, E)$, with non-negative edge costs $c \in \mathbb{R}^E$, the *k-edge connected subgraph problem* (kEC), $k \geq 1$, is the special case of the SNDP where $r_i = k$ for all $i \in V$. In particular, it is the problem of finding a minimum cost k-edge connected spanning subgraph of G, $k \geq 1$. Note that 1EC is the well known minimum cost spanning tree problem. A kEC problem with metric edge costs will be referred to as *metric kEC*.

If multiple copies of the edges are allowed in the solution of kEC, the problem will be referred to as the *multi-k-edge connected subgraph problem* (multi-kEC). We call instances of multi-SNDP where $r_i \in \{0, k\}$, $i \in V$, the *Steiner multi-k-edge connected subgraph problem* (Steiner multi-kEC). Problems kEC and multi-kEC (and Steiner multi-kEC) are special cases of SNDP and multi-SNDP, respectively. In this paper, we examine multi-SNDP, and apply our results to SNDP, metric SNDP, multi-kEC, kEC, and Steiner multi-kEC.

The SNDP is an NP-hard problem, since the Steiner tree problem, which has been shown to be NP-hard [14], is the special case where $r_i \in \{0, 1\}$ for all $i \in V$. Thus, multi-SNDP is an NP-hard problem. Both kEC and multi-kEC ($k \geq 2$) are also known to be NP-hard problems [10]. (The minimum cost spanning tree problem is not NP-hard.) 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* or *performance 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 non-negative, and all not identically

0, objective cost functions. If it can be proved that this largest ratio has value k, the linear programming relaxation is said to have a k *integrality gap*. Furthermore, if the proof of this is constructive and of polynomial time, then it forms a k-approximation algorithm for the problem P.

The best general constant factor approximation guarantee that has existed for multi-SNDP, SNDP, kEC and multi-kEC has, to this point, been 2. This guarantee is due to Kamal Jain [13], who, in 2001, presented a 2-approximation algorithm for finding, in a general graph that may not be simple and has non-negative edge costs, a minimum cost subgraph having at least a specified number of edges in each cut [13]. Initially, multiple copies of edges are not allowed in the solution, however, as is noted near the end of his paper, the 2-approximation algorithm holds as well for the case when multiple copies of edges are allowed in the solution. In addition to providing an approximation guarantee, Jain's algorithm provides a constructive proof that the integrality gap of the linear programming relaxation of the general problem he considers is at most 2. In particular, Jain's result implies that the integrality gap of the LP relaxation of kEC and multi-kEC is ≤ 2.

Goemans and Bertsimas [11] in 1993 presented an approximation algorithm (the improved tree heuristic) for multi-SNDP that, given distinct vertex connectivity requirements $\rho_o = 0 < \rho_1 < \rho_2 < \ldots < \rho_q$, has approximation guarantee $\sum_{j=1}^{q} \frac{f(\rho_j - \rho_{j-1})}{\rho_j}$, where $f(c) = \frac{3}{2}c$ when c is even, and $f(c) = \frac{3c+1}{2}$ when c is odd [11]. The proof used to obtain this is rather complex and involved, and, although not indicated, implicitly yields an upper bound for the integrality gap as well. As a special case, this approximation algorithm has an approximation guarantee (and integrality gap) for multi-kEC of $\frac{3}{2}$ when k is even, and $\frac{3}{2} + \frac{1}{2k}$ when k is odd.

We present an upper bound for the integrality gap for the linear programming relaxation of multi-SNDP; that is, we show that the integrality gap for the linear programming relaxation of the multi-survivable network design problem is at most $\frac{3}{2}\frac{r_m}{r_l}$ when r_m is even, and at most $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ when r_m is odd; where r_m is the maximum edge connectivity requirement induced by the vertex connectivity requirements, and r_l is the smallest nonzero vertex connectivity requirement. We present as well an approximation algorithm for multi-SNDP which has a performance guarantee the same as this upper bound. These are both improvements for the best known bounds for certain special cases of the problem. We show the application of these results (integrality gap and approximation algorithm) to SNDP, metric SNDP, multi-kEC, kEC, metric kEC, and Steiner multi-kEC. Additionally, we discuss the relationship between our approximation algorithm and the improved tree heuristic of Goemans and Bertsimas [11].

Our results for the integrality gap and the approximation algorithms for multi-SNDP and multi-kEC depend upon upper bounds for the optimal values of the minimum cost T-join and the minimum cost spanning tree in terms of the optimal value of the LP relaxation of multi-SNDP, which are also presented.

3.1.2 Related Problems and Overview

Given a general graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \{0, k\}$ for all $i \in V$, $k \geq 1$, the *Steiner k-edge connected subgraph problem* (Steiner kEC) is that of finding a minimum k-edge connected subgraph of G that spans the set of terminal vertices $S = \{i \in V \mid r_i = k\} \subseteq V$; i.e., such that there are at least $r_{ij} = \min(r_i, r_j)$ edge-disjoint paths between all pairs of vertices $i, j \in V$. This problem is NP-hard [17]. Problem kEC is a special case of Steiner kEC, which is a special case of SNDP. If G is a complete graph and multiple copies of the edges are allowed in the solution of Steiner kEC, the problem will be referred to as the *Steiner k-edge connected multi-subgraph problem* (Steiner multi-kEC). Problem multi-kEC is a special case of Steiner multi-kEC, which is a special case of multi-SNDP.

Problem kEC is a special case of the *graph augmentation problem*: Given a complete graph $G = (V, E)$, a subgraph (V, F), $F \subset E$, and a non-negative cost function $c \in \mathbb{R}^E$, find a set of edges $E' \subseteq E \backslash F$ of minimum cost such that the "augmented graph" $(V, F \cup E')$ satisfies a given property [5]. Among other things, this problem can be used for improving the reliability (or survivability) of already existing networks. For kEC we have $F = \emptyset$, and the property to be satisfied is that the augmented graph $(V, F \cup E') = (V, E')$ must be k-edge connected. As is mentioned earlier, kEC is a special case of SNDP and multi-kEC is a special case of multi-SNDP.

For a summary of the current known results on the different versions of kEC, see [15].

The remainder of this paper is organized as follows. In Section 3.2, we give some required background information on minimum cost perfect matchings, T-joins, and spanning trees. In Section 3.3, we present the main results of the paper, along with new upper bounds for the optimal values of the minimum cost T-join and the minimum cost spanning tree in terms of the optimal value of the LP relaxation of multi-SNDP. We also present a family of examples where our integrality gap for the LP relaxation of multi-SNDP, and our approximation algorithm for multi-SNDP, are improvements for the best known bounds of the problem. In Section 3.4, we show the applications of our results to SNDP and metric SNDP. In Section 3.5, we show the applications of our results to multi-kEC, kEC, metric kEC, and Steiner multi-kEC, and discuss the relationship of our approximation algorithm for multi-kEC to Goemans and Bertsimas' improved tree heuristic.

3.2 Minimum Cost Perfect Matchings, T-joins, and Spanning Trees

Before presenting our main results, the following background information is required.

A *minimum cost perfect matching* in a connected graph $G = (V, E)$ with non-negative edge costs, is a minimum cost subset of edges of E such that each vertex in V is incident to exactly one edge in the subset. Using an efficient implementation of Edmonds' Blossom Algorithm, a minimum cost perfect matching in G can be found in $O(n(m+n\log n))$ time [3], where n is the number of vertices and m is the number of edges in G. If G is a complete graph, this becomes $O(n^3)$.

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 $v \in T$. Note that a connected graph has a T-join if and only if $|T|$ is even [2].

Let x_e represent the number of times each edge $e \in E$ is included in the solution of a given problem 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 following theorem states that we can find the cost of an optimal T-join by solving a linear program that has no integer constraints.

Theorem 1 (Cook et. al. [2], 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*

$$\begin{aligned}
&\textit{minimize} \;\; cx && && \qquad (3.1)\\
&\textit{subject to} \quad x(\delta(S)) \;\geq\; 1, && \textit{for all } S \subseteq V \textit{ s.t.} &&\\
& && |S \cap T| \textit{ odd}, && \qquad (3.2)\\
& x_e \;\geq\; 0, && \textit{for all } e \in E. && \qquad (3.3)
\end{aligned}$$

A solution to the minimum cost T-join problem can be found in polynomial time in the following way: Find a shortest (i.e. minimum cost) path, P_{uv}, in G between each pair of vertices $u, v \in T$; and let f_{uv} be the cost of this path. Next, form a complete graph $\hat{G} = (T, \hat{E})$ using the vertices in T and the edge costs f_{uv}, and find a minimum cost perfect matching $\hat{M} \subseteq \hat{E}$ in it. The symmetric difference of the edge-sets of the corresponding paths P_{uv} for $uv \in \hat{M}$, is a minimum cost T-join of G

[2, 4]. The shortest path between two vertices in a connected graph with non-negative edge costs can be determined in $O(n \log n + m)$ time [16]. For a complete graph, this becomes $O(n^2)$ time. Thus, finding a shortest path between each pair of vertices in a complete graph (i.e., finding all pairs shortest paths) takes, in the worst case, $O(n^3)$ time. Therefore, finding a minimum cost T-join can be done, in the worst case, in $O(n^3 + n^3) = O(n^3)$ time.

A *minimum cost spanning tree* (MST) of a complete graph $G = (V, E)$ is a tree of G that includes all the vertices of G (i.e., a spanning tree of G) and has minimum cost. Such a spanning tree can be found in $O(n \log n + m)$ time [6]. This becomes $O(n^2)$ for a complete graph.

Before presenting a corollary that is foundational to our results, we need the following definitions and proposition.

Given a graph $G = (V, E)$, a partition $P = (V_1, V_2, \ldots, V_{k_P})$ of the vertex set V is *feasible* if each part V_i induces a connected subgraph $G[V_i]$ for all $i = 1, 2, \ldots, k_P$. The graph G_P is the *shrunk graph* of G with respect to P, obtained by identifying all the vertices in V_i into a vertex $\tilde{v}_i$ (for each i) and deleting all the edges of G that have both endpoints in the same part, V_i, of the partition P. G_P has k_P vertices, and its edge set, E_P, contains all the edges of G that have endpoints in different parts, V_i, of the partition P.

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. Given a connected graph $G = (V, E)$, consider the collection of spanning trees of G. The incidence vector, x, corresponding to a spanning tree, T, has $x_e = 1$ if T contains e, and $x_e = 0$ otherwise, for all $e \in E$. Let $m = |E|$.

Proposition 2 (Chopra [1]) *Given a connected graph $G = (V, E)$, the dominant* *of the spanning tree polytope is*

$$\left\{ x \in \mathbb{R}^m_{\geq 0} \;:\; \sum_{e \in E_P} x_e \geq k_P - 1,\ \text{for all feasible partitions } P \text{ of } V \right\}. \tag{3.4}$$

Notice that for a complete graph, all partitions of G are feasible.

Chopra [1] shows that, given a partition P of G, the partition inequality $\sum_{e \in E_P} x_e \geq k_P - 1$ defines a facet of (3.4) if and only if the shrunk graph G_P is two-vertex connected. In particular, this means that for a complete graph all the partition inequalities of (3.4) are facet-inducing and thus necessary in the description.

The following corollary is foundational to our results, and follows directly from Proposition 2. It states that we can find the cost of a minimum cost spanning tree by solving a linear program that has no integrality constraints.

Corollary 3 *If $G = (V, E)$ is a connected 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*

$$\begin{aligned} &\textit{minimize} \quad cx && (3.5)\\ &\textit{subject to} \quad \sum_{e \in E_P} x_e \;\geq\; k_P - 1, \quad \text{for all feasible partitions } P \text{ of } V, && (3.6)\\ &\qquad\qquad\quad x_e \;\geq\; 0, \quad \text{for all } e \in E. && (3.7) \end{aligned}$$

Proof: Since the feasible region of constraints (3.6) and (3.7) is precisely the dominant of the spanning tree polytope (Proposition 2), its extreme points are precisely incidence vectors of spanning trees of G. Since LP(3.5) is minimizing over this region, there exists an optimal solution that is the incidence vector of a minimum cost spanning tree. The result follows. ■

3.3 Integrality Gap and Approximation Algorithm

Problem multi-SNDP can be formulated as an integer linear program (ILP) as follows [13]:

$$\begin{aligned} \text{minimize} \quad & cx && (3.8)\\ \text{subject to} \quad x(\delta(S)) & \geq \max_{ij\in\delta(S)} r_{ij}, \quad \text{for all } \emptyset \subset S \subset V, && (3.9)\\ x_e & \geq 0, \quad \text{for all } e \in E, && (3.10)\\ & x_e \quad \text{integer.} \end{aligned}$$

The linear programming (LP) relaxation of multi-SNDP is:

$$\begin{aligned} \text{minimize} \quad & cx && (3.11)\\ \text{subject to} \quad x(\delta(S)) & \geq \max_{ij\in\delta(S)} r_{ij}, \quad \text{for all } \emptyset \subset S \subset V, && (3.12)\\ x_e & \geq 0, \quad \text{for all } e \in E. && (3.13) \end{aligned}$$

The following notation and definitions will be used throughout the rest of this paper: Given any multi-SNDP problem, i.e., given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$, let $r_m = \max\{r_{ij} \mid i, j \in V\}$, $r_l = \min\{r_{ij} \mid r_{ij} > 0,\ i, j \in V\}$, and $V^+ = \{i \in V \mid r_i > 0\}$. Let $G^+ = (V^+, E_{V^+})$ be the complete graph with vertex set V^+, respective vertex connectivity requirements from G, edge set E_{V^+} consisting of the edges E that have both ends in V^+, and edge costs c^+_{ij}, $i, j \in V^+$, the cost of a minimum-cost i-j path in G. Furthermore, let $opt(\textit{multi-SNDP})$ be the cost of an optimal multi-SNDP solution (i.e., the optimal value of the integer linear program (3.8)) on G, and let $opt(\textit{multi-SNDP}_{LP})$ be the cost of an optimal solution to the LP relaxation of multi-SNDP (i.e., the optimal value of the linear program (3.11)) on G.

Given a complete graph $G = (V, E)$ with non-negative edge costs c, the *metric completion*, $G^\triangle = (V, E)$, of G is obtained by setting the edge cost $c^\triangle_{ij}$ to be that of a minimum-cost path connecting vertices i and j in G, for every edge $ij \in E$. Clearly, the edge costs $c^{\triangle}$ are metric. Note that, given a complete graph G' with metric edge costs, its metric completion does not necessarily have the same edge costs (although both have metric edge costs). Clearly, G^+ is $G^{\triangle}$ with the vertices $\{i \in V \mid r_i = 0\}$ removed along with their incident edges. Thus, G^+ is a subgraph of $G^{\triangle}$.

Notice that, for $\emptyset \subset S \subset V^+$, the right hand sides of the constraints (3.12) are the same for G^+ and $G^{\triangle}$ (and for G). This is true since $r_i = 0$ for all $i \in V \backslash V^+$, and thus $\max_{j \in V^+ \backslash S} r_j = \max_{j \in V \backslash S} r_j$, for all $\emptyset \subset S \subset V^+$. Clearly, $\max_{i \in S} r_i$ remains the same. Given a graph H, let $\delta_H(S)$ be the cut induced by the set $S \subseteq V(H)$. Then, for $\emptyset \subset S \subset V^+$,

$$\begin{aligned} \max_{ij \in \delta_{G^+}(S)} r_{ij} &= \min\{\max_{i \in S} r_i, \max_{j \in V^+ \backslash S} r_j\} \\ &= \min\{\max_{i \in S} r_i, \max_{j \in V \backslash S} r_j\} \\ &= \max_{ij \in \delta_{G^{\triangle}}(S)} r_{ij} \quad [= \max_{ij \in \delta_G(S)} r_{ij}]. \end{aligned}$$

Theorem 4 ***(Theorem 3 Goemans and Bertsimas [11])*** *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$ and a non-negative connectivity requirement r_{ij} for every (unordered) pair of vertices $i, j \in V$. Let $G^{\triangle} = (V, E)$ be the metric completion of G, with metric costs $c^{\triangle} \in \mathbb{R}^E$. Then the optimal values of multi-SNDP on G and on $G^{\triangle}$ are equal (and the optimal values of their linear programming relaxations are equal).*

When Theorem 4 is applied to multi-kEC, $k \geq 1$, the following corollary is obtained.

Corollary 5 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and the metric completion $G^{\triangle}$ of G,*

$$opt(\text{multi-}kEC \text{ on } G) = opt(\text{multi-}kEC \text{ on } G^{\triangle}).$$

Lemma 6 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$,*

$$opt(\text{multi-}SNDP) \;\leq\; opt(SNDP).$$

Also,

$$opt(\text{multi-}SNDP_{LP}) \;\leq\; opt(SNDP_{LP}).$$

Proof: The problem multi-SNDP is a relaxation of SNDP (relaxing the constraints that each edge can be chosen at most once in the solution). Similarly, the linear programming relaxation of multi-SNDP is a relaxation of the linear programming relaxation of SNDP. ∎

The following proposition can be obtained directly by combining Lemma 6 with Theorem 4.

Proposition 7 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, let $G^\triangle = (V, E)$ be the metric completion of G, with metric costs $c^\triangle \in \mathbb{R}^E$. Then*

$$opt(\text{multi-}SNDP \text{ on } G) \;\leq\; opt(SNDP \text{ on } G^\triangle)$$

and

$$opt(\text{multi-}SNDP_{LP} \text{ on } G) \;\leq\; opt(SNDP_{LP} \text{ on } G^\triangle).$$

The following theorem follows directly from the parsimonious property of Goemans and Bertsimas [11].

Theorem 8 *Given a complete graph $G = (V, E)$ with metric edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. The optimal value of the LP relaxation of multi-SNDP, LP (3.11), is equal to the optimal value of*

$$\text{minimize} \quad cx \tag{3.14}$$

$$\text{subject to} \quad x(\delta(S)) \;\geq\; \max_{ij \in \delta(S)} r_{ij}, \quad \text{for all } \emptyset \subset S \subset V, \tag{3.15}$$

$$x(\delta(i)) \;=\; 0, \quad \text{for all } i \in V \backslash V^+, \tag{3.16}$$

$$x_e \;\geq\; 0, \quad \text{for all } e \in E. \tag{3.17}$$

Proposition 9 below shows that the optimal value of LP (3.11) is the same for G^+, $G^\triangle$, and G.

Proposition 9 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$,*

$$\begin{aligned} opt(\textit{multi-}SNDP_{LP} \textit{ on } G^+) &= opt(\textit{multi-}SNDP_{LP} \textit{ on } G^\triangle) \\ &= opt(\textit{multi-}SNDP_{LP} \textit{ on } G). \end{aligned}$$

Proof: Let $G^\triangle$ with cost function $c^\triangle$ be the metric completion of G. Let x^* be an optimal solution to LP(3.14) on $G^\triangle$. Let y be x^* restricted to G^+, with edge costs c^+ as defined at the beginning of the section. Since x^* satisfies the constraints (3.16), i.e., since $x^*_e = 0$ on all the edges incident to vertices i with $r_i = 0$, we have that $c^+y = c^\triangle x^*$. By Theorem 8, $c^\triangle x^* = opt(\textit{multi-}SNDP_{LP} \textit{ on } G^\triangle)$.

Clearly, y satisfies the LP relaxation of multi-SNDP (LP(3.11)) on G^+. Suppose that y is not an optimal solution of LP(3.11) on G^+. Let w be an optimal solution of LP(3.11) on G^+. Then $c^+w < c^+y$. Form a solution w' on $G^\triangle$ by setting $w' = 0$ on all the edges that are incident to a vertex $i \in V \backslash V^+$, and $w' = w$ otherwise. Note that w' is a solution to LP(3.14). Then $c^\triangle w' = c^+w < c^+y = c^\triangle x^*$. This is a contradiction, since x^* is an optimal solution to LP(3.14). Thus, y is an optimal solution of LP(3.11) on G^+, i.e., $c^+y = opt(\textit{multi-}SNDP_{LP} \textit{ on } G^+)$. Hence, we have that

$$opt(\textit{multi-}SNDP_{LP} \textit{ on } G^+) = c^+y = c^\triangle x^* = opt(\textit{multi-}SNDP_{LP} \textit{ on } G^\triangle).$$

From Theorem 4, $opt(\textit{multi-}SNDP_{LP} \textit{ on } G^\triangle) = opt(\textit{multi-}SNDP_{LP} \textit{ on } G)$. The result follows. ■

Let $opt(T\textit{-join})$ be the cost of a minimum cost T-join on G.

Lemma 10 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. Let $T \subseteq V^+$, $|T|$ even. The following holds on G:*

$$opt(T\text{-}join) \;\leq\; \frac{1}{r_l}\, opt(multi\text{-}SNDP_{LP}) \;\leq\; \frac{1}{r_l}\, opt(multi\text{-}SNDP).$$

Proof: Let x^* be an optimal solution to the multi-SNDP linear programming relaxation LP(3.11). In particular, x^* satisfies the constraints (3.12). Notice that $\max_{ij \in \delta(U)} r_{ij} \geq 0$, and hence $x^*(\delta(U)) \geq 0$, for all $\emptyset \subset U \subset V$. We show that $\frac{1}{r_l}x^*$ satisfies the constraints (3.2), i.e., that $\frac{1}{r_l}x^*$ satisfies $x(\delta(U)) \geq 1$ for all $\emptyset \subset U \subset V$ such that $|U \cap T|$ is odd. We do this by examining the following two cases for the set $\emptyset \subset U \subset V$:

Case 1: Either $r_i = 0$ for all $i \in U$, or $r_j = 0$ for all $j \in V \backslash U$, or both. Thus, we have that either $\max_{i \in U} r_i = 0$, or $\max_{i \in V \backslash U} r_i = 0$, or both. Notice that, in a complete graph,

$$\max_{ij \in \delta(U)} r_{ij} = \min\{\max_{i \in U} r_i,\ \max_{j \in V \backslash U} r_j\}.$$

Then $\max_{ij \in \delta(U)} r_{ij} = 0$ and thus $x^*(\delta(U)) \geq 0$. Without loss of generality, assume that $r_i = 0$ for all $i \in U$. Since there does not exist a vertex $i \in T$ such that $r_i = 0$, we have that $T \subseteq V \backslash U$. Thus, we have that $|(V \backslash U) \cap T| = |T|$ is even, and $|U \cap T| = 0$ is even. Thus, for all $\emptyset \subset U \subset V$ for which either $r_i = 0$ for all $i \in U$, or $r_j = 0$ for all $j \in V \backslash U$, or both, then $\frac{1}{r_l}x^*$ satisfies $x(\delta(U)) \geq 0$ and $|U \cap T|$ is even.

Case 2: There exists $i \in U$, $j \in V \backslash U$ such that $r_i > 0$ and $r_j > 0$. Thus, $r_{ij} > 0$ and x^* satisfies $x(\delta(U)) \geq r_{ij}$. Since

$$r_l = \min\{r_i \mid r_i > 0,\ i \in V\} = \min\{r_{ij} \mid r_{ij} > 0,\ i, j \in V,\ i \neq j\},$$

therefore $r_{ij} \geq r_l$. Therefore x^* satisfies $x(\delta(U)) \geq r_l$, and hence $\frac{1}{r_l}x^*$ satisfies $x(\delta(U)) \geq 1$.

Thus, $\frac{1}{r_l}x^*$ satisfies the constraints (3.2) for all $\emptyset \subset U \subset V$.

In addition, since x^* satisfies the constraints (3.13), $\frac{1}{r_l}x^*$ also satisfies the inequalities $x_e \geq 0$ for all $e \in E$. Thus, $\frac{1}{r_l}x^*$ satisfies the constraints (3.2) and (3.3). This means that $\frac{1}{r_l}x^*$ is a feasible solution of LP(3.1). Using Theorem 1, we therefore have that, for $T \subseteq V$, $|T|$ even,

$$opt(T\text{-}join) \ \leq \ \frac{1}{r_l}\, cx^* \ = \ \frac{1}{r_l}\, opt(multi\text{-}SNDP_{LP}) \ \leq \ \frac{1}{r_l}\, opt(multi\text{-}SNDP).$$

■

Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, let $opt(MST)$ be the cost of a minimum cost spanning tree (i.e., the optimal value of LP(3.5)) on G.

Lemma 11 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$, the following holds:*

$$opt(MST\ on\ G^+) \ \leq \ \frac{2}{r_l}\, opt(\text{multi-SNDP}_{LP}\ on\ G) \ \leq \ \frac{2}{r_l}\, opt(multi\text{-}SNDP\ on\ G).$$

Proof: Let $G^{\triangle}$ with cost function $c^{\triangle}$ be the metric completion of G. Let x^* be an optimal solution to LP(3.14) on $G^{\triangle}$. Let y be x^* restricted to G^+, with edge costs c^+ (G^+, c^+ as defined at the beginning of this section). Since x^* satisfies the constraints (3.16), i.e., since $x^*_e = 0$ on all the edges incident to vertices i with $r_i = 0$, we have that

$$c^+ y = c^{\triangle} x^* \tag{3.18}$$

and

$$y(\delta(S)) \ \geq \ \max_{ij \in \delta(S)} r_{ij}, \quad \text{for all } \emptyset \subset S \subset V^+, \tag{3.19}$$

$$y_e \ \geq \ 0, \quad \text{for all } e \in E_{V^+}. \tag{3.20}$$

Since y satisfies the constraints (3.19) and since, for all $\emptyset \subset S \subset V^+$, $\max_{ij\in\delta(S)} r_{ij} \geq r_l$, therefore $\frac{2}{r_l}y$ satisfies

$$y(\delta(S)) \ \geq \ \frac{2}{r_l}(\max_{ij \in \delta(S)} r_{ij}) \ \geq \ 2, \qquad \text{for all } \emptyset \subset S \subset V^+. \tag{3.21}$$

Let $P = (S_1, S_2, \ldots, S_{k_P})$ be a partition of G^+. Note that P is always feasible since G^+ is a complete graph. Using (3.21) above, we have, for $\frac{2}{r_l}y$,

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

Thus, $\frac{2}{r_l}y$ satisfies $\sum_{e \in E_P} y_e \geq k_P - 1$, for all feasible partitions $P = (S_1, S_2, \ldots, S_{k_P})$ of G^+.

Note that y also satisfies the constraints (3.20). Thus, $\frac{2}{r_l}y$ satisfies the constraints (3.6) and (3.7) for the graph (V^+, E_{V^+}). This means that $\frac{2}{r_l}y$ is a feasible solution of LP(3.5) on (V^+, E_{V^+}). Using Corollary 3 and equation (3.18),

$$\begin{aligned}opt(MST\ on\ G^+) &\leq \frac{2}{r_l}\, c^{\triangle, V^+} y \\ &= \frac{2}{r_l}\, c^{\triangle} x^*\end{aligned}$$

Applying Theorems 8 and 4, we therefore have that

$$\begin{aligned}opt(MST\ on\ G^+) &= \frac{2}{r_l}\, opt(multi\text{-}SNDP_{LP} \text{ on } G^{\triangle}) \\ &= \frac{2}{r_l}\, opt(multi\text{-}SNDP_{LP} \text{ on } G) \\ &\leq \frac{2}{r_l}\, opt(multi\text{-}SNDP \text{ on } G).\end{aligned}$$

■

Lemmas 10 and 11 will be used to create a solution for multi-SNDP within a given bound of the optimal. Before doing this, we need to define two special graphs. These two graphs will be used for the remainder of this section. Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. Vertices with $r_i = 0$ are called *Steiner vertices*. Let $M = (V, F)$ be a minimum cost spanning tree of G^+. Let $V^{odd} \subset V$ be the set of vertices of G^+ having odd degree in M. Let $J \subseteq E$ be a minimum cost T-join on G^+, where $T = V^{odd}$. Let E' be the edge set consisting of $\lceil \frac{r_m}{2} \rceil$ copies of the edges in F, and J' be the edge set consisting of $\lfloor \frac{r_m}{2} \rfloor$ copies of the edges in J. Let $K^+ = (V, E' \cup J')$. Let K be the multi-subgraph of G that is formed from K^+ by adding the Steiner vertices of G to K^+ and by replacing the edges of K^+ by the minimum-cost paths in G that gave rise to their edge costs. It will be seen at the end of this section that K is the heuristic solution to multi-SNDP that is returned by our approximation algorithm.

The following proposition is needed:

Proposition 12 *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. The graph K^+ is a r_m-edge connected spanning multi-subgraph of G^+, the graph K is a r_m-edge connected multi-subgraph of G, and the total cost of K is equal to the total cost of K^+.*

Proof: We show first that K^+ is a r_m-edge connected spanning multi-subgraph of G^+. Since a connected graph has a T-join if and only if $|T|$ is even [2], and it is known that every graph has an even number of odd degree vertices, therefore, G^+ has a V^{odd}-join.

Clearly, all the vertices of the subgraph $H = (V, F \cup J)$ of G^+ have even degree (note that H may contain multiple copies of the edges). This means that H is an Eulerian graph, i.e., it contains an Euler tour. Hence, H is two-edge connected, i.e., it has two edge disjoint paths between every pair of vertices.

Thus, when r_m is even, K^+ partitions into $\frac{r_m}{2}$ Eulerian subgraphs, i.e., $\frac{r_m}{2}$ 2-edge connected subgraphs. Thus, K^+ has $\frac{r_m}{2} \cdot 2$ pairwise edge-disjoint paths (counting parallel edges as distinct edges) between each pair of vertices, and is thus r_m-edge connected. When r_m is odd, K^+ partitions into $\frac{r_m - 1}{2}$ Eulerian subgraphs and one minimum spanning tree, i.e., $\frac{r_m-1}{2}$ 2-edge connected subgraphs and one connected subgraph. Thus, K^+ has $\frac{r_m-1}{2} \cdot 2 + 1 = r_m$ pairwise edge-disjoint paths (counting parallel edges as distinct edges) between each pair of vertices, and is thus r_m-edge connected. Hence, K^+ is a r_m-edge connected spanning multi-subgraph of G^+.

Form the multi-subgraph, K, of G from K^+ by adding the Steiner vertices of G to K^+ and by replacing the edges of K^+ by the minimum-cost paths in G that gave rise to their edge costs. For each pair of distinct vertices in V^+, the number of pairwise edge-disjoint paths in K is clearly the same as in K^+, hence K is r_m-edge connected. By construction, the total cost of K is equal to the total cost of K^+. ■

Lemmas 10 and 11 lead to the following theorem:

Theorem 13 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. The integrality gap for the linear programming relaxation of multi-SNDP is at most $\frac{3}{2}\frac{r_m}{r_l}$ when r_m is even, and at most $\frac{3r_m+1}{2r_l} = \frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ when r_m is odd.*

Proof: Let $cost(K)$ be the cost of K, and $cost(K^+)$ be the cost of K^+. Let $opt(M \text{ on } G^+)$ be the cost of a minimum cost spanning tree of G^+, and let $opt(J \text{ on } G^+)$ be the cost of an optimal V^{odd}-join on G^+. Using Lemma 10 and Proposition 9,

$$\begin{aligned} opt(J \text{ on } G^+) &\leq \frac{1}{r_l}\, opt(\text{multi-SNDP}_{LP} \text{ on } G^+) \\ &= \frac{1}{r_l}\, opt(\text{multi-SNDP}_{LP} \text{ on } G). \end{aligned}$$

If r_m is even, then $\lceil \frac{r_m}{2} \rceil = \lfloor \frac{r_m}{2} \rfloor = \frac{r_m}{2}$, and, using Proposition 12 and Lemma 11 we have,

$$\begin{aligned} cost(K) = cost(K^+) &= \frac{r_m}{2} \cdot opt(M \text{ on } G^+) + \frac{r_m}{2} \cdot opt(J \text{ on } G^+) \\ &\leq \frac{r_m}{2} \cdot \frac{2}{r_l}\, opt(\text{multi-SNDP}_{LP} \text{ on } G) \\ &\qquad + \frac{r_m}{2} \cdot \frac{1}{r_l}\, opt(\text{multi-SNDP}_{LP} \text{ on } G) \\ &= \frac{3}{2}\frac{r_m}{r_l}\, opt(\text{multi-SNDP}_{LP} \text{ on } G). \end{aligned} \tag{3.22}$$

Therefore, on G,

$$\mathit{opt}(\text{multi-SNDP}) \;\leq\; \mathit{cost}(K) \;\leq\; \frac{3}{2}\frac{r_m}{r_l}\,\mathit{opt}(\text{multi-SNDP}_{LP}),$$

i.e.,

$$\frac{\mathit{opt}(\text{multi-SNDP})}{\mathit{opt}(\text{multi-SNDP}_{LP})} \;\leq\; \frac{3}{2}\frac{r_m}{r_l}. \tag{3.23}$$

Thus, when r_m is even, the integrality gap for the linear programming relaxation of multi-SNDP is at most $\frac{3}{2}\frac{r_m}{r_l}$.

Similarly, if r_m is odd, then $\lfloor\frac{r_m}{2}\rfloor = \frac{r_m-1}{2}$, $\lceil\frac{r_m}{2}\rceil = \frac{r_m+1}{2}$, and we have

$$\begin{aligned}
\mathit{cost}(K) = \mathit{cost}(K^+) &= \frac{r_m+1}{2}\cdot \mathit{opt}(M \;\; on \;\; G^+) + \frac{r_m-1}{2}\cdot \mathit{opt}(J \;\; on \;\; G^+)\\
&\leq \frac{r_m+1}{2}\cdot\frac{2}{r_l}\,\mathit{opt}(\mathit{multi\text{-}SNDP}_{LP} \;\; on \;\; G)\\
&\qquad\qquad + \frac{r_m-1}{2}\cdot\frac{1}{r_l}\,\mathit{opt}(\mathit{multi\text{-}SNDP}_{LP} \;\; on \;\; G)\\
&= \left(\frac{3r_m+1}{2r_l}\right)\,\mathit{opt}(\mathit{multi\text{-}SNDP}_{LP} \;\; on \;\; G)\\
&= \left(\frac{3}{2}\frac{r_m}{r_l}+\frac{1}{2r_l}\right)\,\mathit{opt}(\mathit{multi\text{-}SNDP}_{LP} \;\; on \;\; G).
\end{aligned} \tag{3.24}$$

Therefore, on G,

$$\mathit{opt}(\text{multi-SNDP}) \;\leq\; \mathit{cost}(K) \;\leq\; \left(\frac{3}{2}\frac{r_m}{r_l}+\frac{1}{2r_l}\right)\,\mathit{opt}(\mathit{multi\text{-}SNDP}_{LP}),$$

i.e.,

$$\frac{\mathit{opt}(\text{multi-SNDP})}{\mathit{opt}(\text{multi-SNDP}_{LP})} \;\leq\; \frac{3}{2}\frac{r_m}{r_l}+\frac{1}{2r_l}. \tag{3.25}$$

Thus, the integrality gap for the linear programming relaxation of multi-SNDP when r_m is odd is at most $\frac{3}{2}\frac{r_m}{r_l}+\frac{1}{2r_l}$. ■

From Theorem 13, we obtain the following two corollaries.

Corollary 14 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. The cost of K is less than or equal to $\frac{3}{2}\frac{r_m}{r_l}$ of the optimal solution value for the LP relaxation of multi-SNDP if r_m is even, and less than or equal to $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ of the optimal solution value for the LP relaxation of multi-SNDP if r_m is odd.*

Proof: By equations (3.22) and (3.24). ■

Corollary 15 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. The cost of K is less than or equal to $\frac{3}{2}\frac{r_m}{r_l}$ of the optimal solution value for multi-SNDP if r_m is even, and less than or equal to $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ of the optimal solution value for multi-SNDP if r_m is odd.*

Proof: We know that $opt(\text{multi-SNDP}_{LP}) \leq opt(\text{multi-SNDP})$. Combining this with equations (3.22) and (3.24) we obtain the result. ■

The constructive proof of Theorem 13, along with Corollary 15, yields the following approximation algorithm for multi-SNDP:

Algorithm A

Input: A complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$.
Output: An r_m-edge connected spanning multi-subgraph (i.e., subgraph with multiple copies of the edges allowed) of G, having cost within $\frac{3}{2}\frac{r_m}{r_l}$ of the optimal solution value for multi-SNDP, if r_m is even, and within $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ of the optimal solution value for multi-SNDP, if r_m is odd.

A0. Form the complete graph G^+ with vertex set V^+, respective vertex connectivity requirements from G, and edge costs c^+_{ij}, $i, j \in V^+$, where c^+_{ij} is the cost of a minimum-cost i-j path in G.

A1. Find a minimum cost spanning tree, $M = (V, F)$, of G^+ and take $\lceil \frac{r_m}{2} \rceil$ copies of each of its edges. Let this form the edge set E'. Let $V^{odd} \subset V$ be the set of vertices of G^+ having odd degree in M.

A2. Find a minimum cost T-join $J \subseteq E$, on G^+, where $T = V^{odd}$, i.e., M has a corresponding V^{odd}-join. This is accomplished by finding a minimum cost "pairing" of the odd degree vertices of M using shortest paths from G^+. Take $\lfloor \frac{r_m}{2} \rfloor$ copies of the edges in J, and let this form the edge set J'.

A3. Combine the minimum cost spanning trees from (A1) and the T-joins from (A2), i.e., form the multi-subgraph $K^+ = (V, E' \cup J')$.

A4. Form a multi-subgraph, K, of G from K^+ by adding the Steiner vertices of G to K^+ and by replacing each edge of K^+ by the corresponding minimum-cost path in G that gave rise to its edge cost.

Theorem 16 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. The above algorithm yields, in polynomial time, a feasible multi-subgraph K which is r_m-edge connected and which has cost less than or equal to $\frac{3}{2}\frac{r_m}{r_l}$ of the optimal solution value for multi-SNDP when r_m is even, and cost less than or equal to $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ of the optimal solution value for multi-SNDP when r_m is odd. Thus, it is a $\frac{3}{2}\frac{r_m}{r_l}$-approximation algorithm for multi-SNDP, for r_m even, and a $(\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l})$-approximation algorithm for multi-SNDP, for r_m odd.*

Proof: From Theorem 13, K is an r_m-edge connected spanning multi-subgraph of G. By Corollary 15, it has cost less than or equal to $\frac{3}{2}\frac{r_m}{r_l} opt$(multi-SNDP) when r_m is even, and cost less than or equal to $(\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}) opt$(multi-SNDP) when r_m is odd. From the discussion in Section 3.2, this algorithm has worst case running time $O(n^3)$. ■

Jain [13] gave both an approximation guarantee of 2 for multi-SNDP, and an integrality gap of 2 for the LP relaxation of multi-SNDP. Given distinct vertex connectivity requirements $\rho_o = 0 < \rho_1 < \rho_2 < \ldots < \rho_q$, Goemans and Bertsimas' [11] improved tree heuristic for multi-SNDP has approximation guarantee $\sum_{j=1}^{q} \frac{f(\rho_j - \rho_{j-1})}{\rho_j}$, where $f(c) = \frac{3}{2}c$ when c is even, and $f(c) = \frac{3c+1}{2}$ when c is odd. Although not indicated, the proof of this implicitly yields an integrality gap of $\sum_{j=1}^{q} \frac{f(\rho_j - \rho_{j-1})}{\rho_j}$.

For r_m even, our approximation guarantee (and the integrality gap of the LP relaxation) for multi-SNDP is strictly less than 2 (and is thus an improvement on Jain's result when applied to multi-SNDP) when $\frac{r_m}{r_l} < \frac{4}{3}$. For r_m odd, our approximation guarantee (and the integrality gap of the LP relaxation) for multi-SNDP is strictly less than 2 when $\frac{r_m}{r_l} + \frac{1}{3r_l} < \frac{4}{3}$.

When $r_m = r_l = k$ (i.e., when we are looking at the special case of multi-kEC), our approximation guarantee and integrality gap, and those of Goemans and Bertsimas are the same – equaling $\frac{3}{2}$ for k even, and $\frac{3}{2} + \frac{1}{2k}$ for k odd. When $r_m > r_l$, there are certain instances where our approximation guarantee and integrality gap is an improvement both on the bounds of Goemans and Bertsimas and on the bounds of Jain, hence providing for those instances a better approximation guarantee for multi-SNDP and integrality gap for the LP relaxation of multi-SNDP than has been previously known. It is difficult to determine all the instances where our bound is an improvement on the best known bounds, however, for an example of a family of such instances, see the family in Section 3.3.1.

Thus, our integrality gap for the LP relaxation of multi-SNDP and our approximation algorithm for multi-SNDP are an improvement on the best known bounds for special cases of multi-SNDP.

3.3.1 A Family of Examples

Recall that Jain [13] presented an approximation guarantee (and upper bound on integrality gap) of 2 for multi-SNDP. Recall also that Goemans and Bertsimas [11] presented an approximation guarantee (and upper bound on integrality gap) for multi-SNDP of $\sum_{j=1}^{q} \frac{f(\rho_j - \rho_{j-1})}{\rho_j}$, where $f(c) = \frac{3}{2}c$ when c is even, and $f(c) = \frac{3c+1}{2}$ when c is odd; and where $\rho_o = 0 < \rho_1 < \rho_2 < \ldots < \rho_q$ are the distinct vertex connectivity requirements.

In this section, we present a family of examples for multi-SNDP where the approximation guarantee for multi-SNDP of Theorem 16, and the integrality gap of the LP relaxation of multi-SNDP of Theorem 13, are both less than Jain's result and less than that of Goemans and Bertsimas (and hence are an improvement on the best known bounds). Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. Let $\mathcal{F}$ be the family of multi-SNDP problems such that:

(i) $1 < r_l < r_m$ if r_m is even; and $1 < r_l < r_m - 1$ if r_m is odd

(ii) $\frac{r_m}{r_l} < \frac{4}{3}$ if r_m is even; and $\frac{r_m}{r_l} + \frac{1}{3r_l} < \frac{4}{3}$ if r_m is odd

(iii) for all $r \in \{r_l, r_l + 1, r_l + 2, \ldots, r_m - 1, r_m\}$, there exists a vertex in G with vertex connectivity requirement r.

Note that under conditions (i)-(iii), the bound of Goemans and Bertsimas is $\frac{\frac{1}{2}[3r_l+1]}{r_l} + \frac{2}{r_l+1} + \frac{2}{r_l+2} + \ldots + \frac{2}{r_m}$ for r_l odd; and $\frac{\frac{3}{2}r_l}{r_l} + \frac{2}{r_l+1} + \frac{2}{r_l+2} + \ldots + \frac{2}{r_m}$ for r_l even.

Recall that our bound (both approximation guarantee and integrality gap) for multi-SNDP is at most $\frac{3}{2}\frac{r_m}{r_l}$ when r_m is even, and $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ when r_m is odd. We now show that on the family $\mathcal{F}$ this is an improvement on the best known bounds for multi-SNDP. First, we will show that condition (ii) ensures that our bound is strictly less than 2 for the family $\mathcal{F}$, and hence an improvement on Jain's bound [13]. This is seen by noting that $\frac{3}{2}\frac{r_m}{r_l} < 2$ if and only if $\frac{r_m}{r_l} < \frac{4}{3}$; and $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l} < 2$ if and only if $\frac{r_m}{r_l} + \frac{1}{3r_l} < \frac{4}{3}$.

Secondly, we will show that conditions (i)-(iii) ensure that our bound is strictly less than that of Goemans and Bertsimas [11]. Note that when $r_m = r_l$, our bound and that of Goemans and Bertsimas is the same, and less than that of Jain. Thus we only examine instances where $r_l < r_m$.

Notice that when $r_l = 1$ and r_m is even, condition (ii) implies that $r_m = 0$, which is a contradiction with $r_l < r_m$. When $r_l = 1$ and r_m is odd, condition (ii) implies that $r_m < 1$, which is a contradiction with $r_l < r_m - 1$. Thus we state that $r_l > 1$. Note that for both r_m even and r_m odd we have, by condition (ii), that $\frac{r_m}{r_l} < \frac{4}{3}$, and thus $\frac{1}{r_m} > \frac{3}{4r_l}$. Furthermore, notice that $r_l < r_l + 1 < r_l + 2 < \ldots < r_m - 1 < r_m$ implies that $\frac{1}{r_l} > \frac{1}{r_l+1} > \frac{1}{r_l+2} > \ldots > \frac{1}{r_m-1} > \frac{1}{r_m}$. To show that conditions (i)-(iii) ensure that our bound is strictly less than that of Goemans and Bertsimas for problems in the family $\mathcal{F}$, we consider two cases.

Case 1: r_l is odd, and r_m is even or odd. Notice that r_l is odd and $r_l < r_m$ automatically implies that $r_l < r_m - 1$ when r_m is odd. The bound of Goemans and Bertsimas [11] for problems in $\mathcal{F}$ with r_l odd is

$$\begin{aligned}
\frac{\frac{1}{2}(3r_l+1)}{r_l} + \frac{2}{r_l+1} + \frac{2}{r_l+2} + \ldots + \frac{2}{r_m} &> \frac{3r_l+1}{2r_l} + \frac{2}{r_m} + \frac{2}{r_m} + \ldots + \frac{2}{r_m} \\
&= \frac{3r_l+1}{2r_l} + (r_m - r_l)\frac{2}{r_m}.
\end{aligned}$$

For both r_m even and r_m odd we have $\frac{1}{r_m} > \frac{3}{4r_l}$, and thus

$$\begin{aligned}
\frac{\frac{1}{2}(3r_l+1)}{r_l} + \frac{2}{r_l+1} + \frac{2}{r_l+2} + \ldots + \frac{2}{r_m} &> \frac{3r_l+1}{2r_l} + (2)(r_m - r_l)\left(\frac{3}{4r_l}\right) \\
&= \frac{3r_l+1}{2r_l} + \frac{3r_m - 3r_l}{2r_l} \\
&= \frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l} \\
&> \frac{3}{2}\frac{r_m}{r_l}.
\end{aligned}$$

Since our bound is at most $\frac{3}{2}\frac{r_m}{r_l}$ when r_m is even, and $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ when r_m is odd, it follows that Goemans and Bertsimas' bound is always greater than our bound for problems in family $\mathcal{F}$ with r_l odd and r_m even or odd.

Case 2a: r_l is even, and r_m is even. The bound of Goemans and Bertsimas [11] for problems in $\mathcal{F}$ with r_l even is

$$\begin{aligned}\frac{\frac{3}{2}r_l}{r_l} + \frac{2}{r_l+1} + \frac{2}{r_l+2} + \ldots + \frac{2}{r_m} &> \frac{3}{2} + \frac{2}{r_m} + \frac{2}{r_m} + \ldots + \frac{2}{r_m} \\ &= \frac{3}{2} + (r_m - r_l)\frac{2}{r_m}.\end{aligned}$$

Applying $\frac{1}{r_m} > \frac{3}{4r_l}$, we have

$$\begin{aligned}\frac{\frac{3}{2}r_l}{r_l} + \frac{2}{r_l+1} + \frac{2}{r_l+2} + \ldots + \frac{2}{r_m} &> \frac{3}{2} + (2)(r_m - r_l)\left(\frac{3}{4r_l}\right) \\ &= \frac{3r_l}{2r_l} + \frac{3r_m - 3r_l}{2r_l} \\ &= \frac{3}{2}\frac{r_m}{r_l}.\end{aligned}$$

Since our bound is at most $\frac{3}{2}\frac{r_m}{r_l}$ when r_m is even, it follows that Goemans and Bertsimas' bound is always greater than our bound for problems in family $\mathcal{F}$ with r_l even and r_m even.

Case 2b: r_l is even, and r_m is odd. Notice that if r_l is even, r_m is odd, and $r_l = r_m - 1$, then our bound is not strictly less than that of Goemans and Bertsimas: Assume it is, i.e., assume that $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l} < \frac{\frac{3}{2}r_l}{r_l} + \frac{2}{r_m} = \frac{3}{2} + \frac{2}{r_m}$. Substituting in $r_l = r_m - 1$, we get $\frac{3r_m+1}{2(r_m-1)} < \frac{3r_m+4}{2r_m}$. Simplifying, this leads to $0 < -8$, which is a contradiction. Thus, we must have that $r_l < r_m - 1$ when r_m is odd.

The bound of Goemans and Bertsimas [11] for problems in $\mathcal{F}$ with r_l even is

$$\begin{aligned}\frac{\frac{3}{2}r_l}{r_l} + \frac{2}{r_l+1} + \frac{2}{r_l+2} + \ldots + \frac{2}{r_m} &> \frac{3}{2} + \frac{2}{r_m-1} + \frac{2}{r_m-1} + \ldots + \frac{2}{r_m-1} + \frac{2}{r_m} \\ &= \frac{3}{2} + (r_m - r_l - 1)\frac{2}{r_m-1} + \frac{2}{r_m}\end{aligned}$$

Since our bound is at most $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ when r_m is odd, it follows that Goemans and Bertsimas' bound is always greater than our bound for problems in family $\mathcal{F}$ with r_l even and r_m odd provided that

$$\frac{3}{2} + (r_m - r_l - 1)\frac{2}{r_m - 1} + \frac{2}{r_m} \quad > \quad \frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}\ , \tag{3.26}$$

i.e., provided that,

$$\frac{7r_m^2 - 3r_m - 4r_m r_l - 4}{2r_m(r_m - 1)} \quad > \quad \frac{3r_m + 1}{2r_l}\ .$$

This is true provided that

$$\frac{7r_m^2 r_l - 3r_m r_l - 4r_m r_l^2 - 4r_l - (3r_m + 1)(r_m^2 - r_m)}{r_m r_l(r_m - 1)} \quad > \quad 0\ ,$$

i.e.,

$$7r_m^2 r_l - 3r_m r_l - 4r_m r_l^2 - 4r_l - 3r_m^3 + 2r_m^2 + r_m \ > \ 0\ .$$

Notice that $r_m > r_l + 1$, r_l even, and r_m odd imply that $r_m = r_l + n$, where $n \in \mathbb{N}$ is odd and $n \geq 3$. Substituting in $r_m = r_l + n$, where $n \geq 3$ is odd, we obtain that the above inequality is equivalent to

$$(n-1)r_l^2 + (-2n^2 + n - 3)r_l + (-3n^3 + 2n^2 + n) \ > \ 0, \quad n \geq 3,\ n \text{ odd}. \tag{3.27}$$

Note that $r_m < \frac{4}{3}r_l - \frac{1}{3}$ (from condition (ii)) and $r_m = r_l + n$, $n \geq 3$ odd, implies that $r_l > 3n + 1$, i.e., that $r_l \geq 3n + 3$ for $n \geq 3$ odd. Using this, it is easily seen that inequality (3.27) is true. Thus, inequality (3.26) is true. Hence, Goemans and Bertsimas' bound is always greater than our bound for problems in family $\mathcal{F}$ with r_l even and r_m odd.

Therefore, Goemans and Bertsimas' bound is always greater than our bound for problems in family $\mathcal{F}$. Hence, for problems in family $\mathcal{F}$, our bound is an improvement on both the bound of Jain and the bound of Goemans and Bertsimas.

3.4 Application of Multi-SNDP to SNDP and Metric SNDP

As application to SNDP, applying Proposition 7 to Lemmas 10 and 11, we obtain the following:

Corollary 17 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, let $G^\triangle = (V, E)$ be the metric completion of G, with metric costs $c^\triangle \in \mathbb{R}^E$. Then*

$$opt(T\text{-}join\ on\ G) \ \leq\ \frac{1}{r_l}\, opt(SNDP_{LP}\ on\ G^\triangle) \leq\ \frac{1}{r_l}\, opt(SNDP\ on\ G^\triangle),$$

and

$$opt(MST\ on\ G) \ \leq\ \frac{2}{r_l}\, opt(SNDP_{LP}\ on\ G^\triangle) \leq\ \frac{2}{r_l}\, opt(SNDP\ on\ G^\triangle).$$

Applying this to metric-SNDP, we obtain

Corollary 18 *Given a complete graph $G' = (V, E)$ with metric costs,*

$$opt(T\text{-}join) \ \leq\ \frac{1}{r_l}\, opt(metric\ SNDP_{LP}) \leq\ \frac{1}{r_l}\, opt(metric\ SNDP),$$

and

$$opt(MST\ on\ G'^{+}) \ \leq\ \frac{2}{r_l}\, opt(metric\ SNDP_{LP}\ on\ G') \leq\ \frac{2}{r_l}\, opt(metric\ SNDP\ on\ G').$$

Setting $r_l = k$, Corollary 17 applies to kEC, and Corollary 18 (with $G'^{+} = G'$) applies to metric kEC.

3.5 Application of Multi-SNDP to Multi-kEC, kEC, and Steiner Multi-kEC

The results for multi-SNDP apply to multi-kEC as follows: Lemmas 10 and 11 hold for multi-kEC. Simply set $r_i = k$ for all $i \in V$. As we will see, the remainder of the results are similar, except that adjustments need to be made based on the fact that the graph G^+ is no longer required (since there are no Steiner vertices).

When considering multi-kEC, we need to define a special graph. Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. Let $M = (V, F)$ be a minimum cost spanning tree of G. Let $V^{odd} \subset V$ be the set of vertices of G having odd degree in M. Let $J \subseteq E$ be a minimum cost T-join on G, where $T = V^{odd}$. Let E' be the edge set consisting of $\lceil \frac{r_m}{2} \rceil$ copies of the edges in F, and J' be the edge set consisting of $\lfloor \frac{r_m}{2} \rfloor$ copies of the edges in J. Let $K = (V, E' \cup J')$. It will be seen that K is the heuristic solution to multi-kEC that is returned by our abbreviated approximation algorithm.

The following proposition is obtained from Proposition 12.

Proposition 19 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, $k \geq 2$, $k \in \mathbb{Z}$. The graph $K = (V, E' \cup J')$ is a k-edge connected spanning multi-subgraph of G.*

Theorem 13 (on integrality gap), Corollary 14, and Corollary 15, all apply to multi-kEC. Simply set $r_m = r_l \equiv k$ and $r_i \equiv k$ for all $i \in V$.

When applying the approximation algorithm for multi-SNDP in Section 3.3 to multi-kEC, $r_m = r_l \equiv k$, and steps A0 and A4 can be omitted. It follows directly from Theorem 16 that this is a $\frac{3}{2}$-approximation algorithm for multi-kEC, for k even, and a $(\frac{3}{2} + \frac{1}{2k})$-approximation algorithm for multi-kEC, for k odd.

Note that the bounds for integrality gap and our approximation algorithm hold as well for Steiner multi-kEC, and, in this case, are strictly less than 2 and equal to the bounds of Goemans and Bertsimas. Hence, they are equal to the best known values to date.

When the edge costs are uniform in a general graph, Gabow et al. [8] give a rounding algorithm that has an approximation guarantee of $1 + \frac{2}{k}$ for kEC when $k > 1$, and provide tight examples for these guarantees. In addition, Gabow et al. [8]

show that the integrality gap of the LP relaxation of kEC with uniform edge costs is at most $1 + \frac{2}{k}$, both for undirected and directed graphs. It is also interesting to note that, for uniform edge costs in a general multi-graph, there is a $\frac{3}{2}$-approximation algorithm for finding a 3-edge connected (spanning) subgraph [7].

Setting $r_l = k$, Corollary 17 applies to kEC, and Corollary 18 applies to metric kEC.

3.6 Conclusion

In this paper, we showed that the integrality gap for the linear programming relaxation of multi-SNDP is at most $\frac{3}{2}\frac{r_m}{r_l}$ when r_m is even, and at most $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ when r_m is odd, where r_m is the maximum vertex connectivity requirement and r_l is the smallest nonzero vertex connectivity requirement. We presented an approximation algorithm for multi-SNDP which has a performance guarantee the same as this upper bound. These are improvements for the best known bounds for certain special cases of the problem, and we presented a family of examples demonstrating this. We also discussed the relationship between our approximation algorithm for multi-kEC and the improved tree heuristic of Goemans and Bertsimas [11].

Our results for integrality gap and the approximation algorithms for multi-SNDP and multi-kEC depend upon new upper bounds for the optimal values of a minimum cost T-join and an MST in terms of the optimal value of the LP relaxation of multi-SNDP, bounds which we also presented.

Lastly, we showed the application of our integrality gap and approximation algorithm results of multi-SNDP to SNDP, metric SNDP, multi-kEC, kEC, metric kEC, and Steiner multi-kEC.

3.7 Acknowledgment

This research was partially supported by grants from the National Sciences and Engineering Research Council of Canada.

Bibliography

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

[2] W.J. Cook, W.H. Cunningham, W.R. Pulleyblank, A. Shrijver, Combinatorial Optimization, Wiley Interscience, 1998 (Section 5.4).

[3] W. Cook, A. Rohe, Computing Minimum Weight Perfect Matchings, Informs Journal on Computing 11 (1999) 138-148.

[4] J. Edmonds, E.L. Johnson, Matching, Euler Tours and the Chinese Postman, Math. Prog. 5 (1973) 88-124.

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

[6] 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.

[7] H.N. Gabow, An Ear Decomposition Approach to Approximating the Smallest 3-edge Connected Spanning Subgraph of a Multigraph, SIAM J. Discrete Math. 18 No.1 (2004) 41-70.

[8] H.N. Gabow, M.X. Goemans, E. Tardos, D.P. Williamson, Approximating the Smallest k-edge Connected Spanning Subgraph by LP-rounding, Proceedings

of the sixteenth annual ACM-SIAM symposium on Discrete algorithms (2005) 562-571.

[9] H.N. Gabow, M.X. Goemans, D.P. Williamson, An Efficient Approximation Algorithm for the Survivable Network Design Problem, Mathematical Programming 82 (1998) 13-40.

[10] 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).

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

[12] M. Grötschel, C.L. Monma, M. Stoer, Design of Survivable Networks, in: M.O. Bell, T.L. Magnant, C.L. Monma, G.L. Nemhauser, eds, Network Models (Handbooks in OR & MS, Vol.7), North-Holland (1995).

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

[14] R.M. Karp, Reducibility Among Combinatorial Problems, in: R.E. Miller, J.W. Thatcher, eds, Complexity of computer computations, Plenum Press (NY) (1972) 85-104.

[15] G. Kortsarz, Z. Nutov, Approximating Minimum-Cost Connectivity Problems, in: T.F. Gonzalez, ed., Handbook of Approximation Algorithms and Metaheuristics, Chapman & Hall (2007).

[16] L. Lovász, et. al., Combinatorics in Computer Science, in: R.L. Graham, M. Grötschel, L. Lovász, eds., Handbook of Combinatorics Volume II, Elsevier Science (1995).

[17] C.L. Monma, B.S. Munson, W.R. Pulleyblank, Minimum-weight Two-connected Spanning Networks, Math. Prog. 46 (1990) 153-171.

Chapter 4

Survivable Network Design Problem with Low Connectivities

• Sylvia Boyd and Amy Cameron. Extensions and Vertex Connectivity, Technical Report, University of Ottawa, 2010.

Work of Amy Cameron under the supervision of Sylvia Boyd. Results are joint work of Amy Cameron and Sylvia Boyd.

Abstract

We examine the special cases of both the survivable network design problem and the vertex connected survivable network design problem (and related variations of these problems) with low vertex connectivity requirements. We investigate cases where the optimal values of the multi-survivable network design problem and the survivable network design problem are equal. We show that there is an upper bound of $\frac{3}{2}$ on the integrality gaps of the linear programming relaxations of both the metric survivable network design problem with vertex connectivity requirements 0,1,2, and the metric vertex-connected survivable network design problem with vertex connectivity requirements 0,1,2. We also present $\frac{3}{2}$-approximation algorithms for both of these problems. Our results are an improvement on the previously best known integrality gap bound and approximation guarantee for the problems, which was 2 [11]. These results also extend the following previously known results: A $\frac{3}{2}$ upper bound on the integrality gap of the linear programming relaxation of the metric 2-edge connected problem [1]; a $\frac{3}{2}$-approximation algorithm for the metric 2-vertex connected problem [12]; and, on an unweighted graph, a $\frac{3}{2}$-approximation algorithm for the vertex-connected survivable network design problem with vertex connectivity requirements 1,2 [26]. Our $\frac{3}{2}$ approximation guarantee and integrality gap bound applies to the Steiner tree problem with either uniform or metric edge costs, and improves the previous best known integrality gap bound for these cases.

4.1 Introduction

An important general problem in network design is the survivable network design problem. Given a network of centres and specific non-negative costs for connecting any two centres with a link, this problem consists of finding a cheapest way to construct a network so that specific parts of the network remain connected even if certain numbers of links are lost. The problem has many practical, cost-saving applications, such as the design of reliable communication and transportation networks, and is used in designing fibre optic telecommunications networks [13, 18]. Here, "cost" can be interpreted as the construction cost, and "reliable" as the continuation of services in the event of a catastrophe causing the loss of a given number of "links" (i.e. communications lines, pipeline segments, or roads). A continuation of services in the event of certain losses is ensured by the network having been designed in such a way that service can be restored by being re-routed through other existing centres and links. For such applications, finding a solution that is either optimal or very close to optimal means substantial financial savings for the company constructing the network. In some applications, it is useful to allow more than one link to be built between a pair of centres in order to build a survivable 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. There are many networks, such as telecommunications networks, whose centres and links have very low failure probabilities [27], thus it is of use to examine the survivable network design problem (and its variations) for special cases when certain parts of the network are connected, and other parts remain connected even if one of the network's links is lost (or, respectively, if one of the network's centres is lost).

4.1.1 Definitions and Problems

More formally, the *(edge connected) survivable network design problem* (SNDP) is defined as follows: Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. Vertices $i \in V$ with vertex connectivity requirement $r_i = 0$ are called *Steiner vertices*. For each pair of distinct vertices $i, j \in V$, let the edge connectivity requirement be $r_{ij} = \min\{r_i, r_j\}$. The problem is to find a minimum cost subgraph that has, for each pair of distinct vertices $i, j \in V$, at least r_{ij} edge-disjoint paths between i and j. In a feasible solution of SNDP, the loss of any $0 < l < \max_{ij \in E} r_{ij}$ edges still allows all vertices with vertex connectivity requirement greater than l to be connected to each other (i.e., for $i, j \in V$, if $r_i > l$ and $r_j > l$, then if any l edges are removed there remains at least one edge-disjoint path joining i and j). When the edge costs $c \in \mathbb{R}^E$ of a graph are non-negative and satisfy the triangle inequality $c_{uw} \leq c_{uv} + c_{vw}$ for all distinct vertices $u, v, w \in V$, the costs are called *metric*. Problem SNDP with metric edge costs will be referred to as *metric SNDP*.

A *multi-subgraph* is a subgraph that may have multiple copies of at least one edge. If multiple copies of the edges are allowed in the solution of SNDP, i.e., a multi-subgraph is allowed, the problem will be referred to as the *(edge-connected) multi-survivable network design problem* (multi-SNDP).

A set of paths between vertices i and j is pairwise *internally vertex-disjoint* if no two of the paths have an internal vertex in common (i.e., the only vertices any of the paths have in common are the vertices i, j). Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$, when the problem is to find a minimum cost subgraph that has, for each pair of distinct vertices $i, j \in V$, at least r_{ij} pairwise internally vertex-disjoint paths between i and j, the problem is referred to as the *vertex connected survivable network design problem* (VC-SNDP). In such a feasible solution, the loss of any $0 < l < \max_{ij \in E} r_{ij}$ vertices still allows all vertices with vertex connectivity requirement greater than l to be connected with each other (i.e., for $i, j \in V$, if $r_i > l$ and $r_j > l$, then if any l vertices are removed there remains at least one internally vertex-disjoint path joining i and j). Problem VC-SNDP is a special case of SNDP, since all feasible solutions to VC-SNDP are also feasible solutions to SNDP, but not vice versa. Thus, the optimal value of VC-SNDP is always at least that of SNDP. Problem VC-SNDP with metric edge costs will be referred to as *metric VC-SNDP*. Generally, multiple copies of edges are not allowed in the solutions of VC-SNDP, since multiple edges do not increase the vertex connectivity. A *cut vertex* is a vertex whose removal disconnects a connected graph. A vertex *cut set* is a subset $U \subset V$ such that $|U| \leq (\max_{ij \in E} r_{ij}) - 1$ and whose removal disconnects the graph.

Unless otherwise stated, in this paper we are assuming that the given G is complete. This can be done without loss of generality, since, if G is not complete, we can make it into a complete graph by simply adding in all the "missing" edges and assigning each of them an appropriately large cost.

A graph is *k-edge connected* (respectively, *k-vertex connected*), $k \geq 1$, $k \in \mathbb{Z}$, 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. All k-vertex connected graphs are k-edge connected, but not vice versa. If a graph is k-vertex connected, then it is j-vertex connected for $1 \leq j < k$, and it has at least $k+1$ vertices. A graph is 2-vertex connected if and only if it has no cut vertices.

The special case of SNDP where $r_i = k$, $k \in \mathbb{Z}_{\geq 1}$, for all $i \in V$ is called the *k-edge connected subgraph problem* (kEC). In particular, kEC is the problem of finding a minimum cost k-edge connected spanning subgraph of G. Notice that 1EC is simply the well known minimum cost spanning tree problem (MST). Problem kEC problem with metric edge costs will be referred to as *metric kEC*. If multiple copies of the edges are allowed in the solution of kEC, the problem will be referred to as the *multi-k-edge* *connected subgraph problem* (multi-kEC). Clearly, kEC and multi-kEC are special cases of SNDP and multi-SNDP, respectively.

Similarly, the special case of VC-SNDP where $r_i = k$, $k \in \mathbb{Z}_{\geq 1}$, for all $i \in V$ is called the *k-vertex connected subgraph problem* (kVC). When multiple copies of edges are allowed in the solution, we denote the problem by *multi-kVC*. Problem kVC with metric edge costs will be referred to as *metric kVC*. Problem 2VC is sometimes called the *biconnected augmentation problem* [12]. It can be shown that the optimal values of metric 2EC and metric 2VC are equal. This is an implicit result of the shortcutting method by Frederickson and Ja'Ja' [12]), and is used by Carr and Ravi [6], and by Monma, Munson, and Pulleyblank [28].

Given a general graph, $G = (V, E)$, with non-negative edge costs $c \in \mathbb{R}^E$, and a subset of vertices $T \subseteq V$, the *Steiner tree problem* (ST) is the problem of finding a minimum cost subgraph that spans all the vertices T. Vertices in the subset T are called *terminal* (or *required*) vertices. The non-terminal vertices $V \backslash T$ are called *Steiner vertices* [20, 23]. A feasible solution to ST includes all the terminal vertices and may include some, all, or no Steiner vertices. Problem ST is the special case of SNDP where $r_i \in \{0, 1\}$, $i \in V$.

The special case of SNDP where $r_i \in \{0, k\}$, $k \in \mathbb{Z}_{\geq 1}$, for all $i \in V$ is called the *Steiner k-edge connected subgraph problem* (Steiner-kEC). In other words, Steiner-kEC is the problem of finding a minimum cost k-edge connected subgraph of G that spans the set of terminal vertices $T = \{i \in V \mid r_i = k\} \subseteq V$; i.e., such that there are at least k edge-disjoint paths between all pairs of distinct vertices $i, j \in S$. Similarly, the special case of multi-SNDP where $r_i \in \{0, k\}$, $k \in \mathbb{Z}_{\geq 1}$, for all $i \in V$ is called the *Steiner k-edge connected multi-subgraph problem* (Steiner multi-kEC). Problems Steiner kEC and Steiner multi-kEC are special cases of SNDP and multi-SNDP, respectively. Furthermore, problems kEC and multi-kEC are the special case of Steiner-kEC and Steiner multi-kEC, respectively, where $T = V$. The related *k-restricted Steiner tree problem* is that of finding a minimum cost Steiner tree whose full components (maximal subtrees whose leaves are all the terminal vertices and whose internal vertices are Steiner vertices) contain at most k terminal vertices.

Problem ST is an NP-hard problem [14, 15, 22]. As well, problem Steiner-2EC with edge costs all 1 (i.e., uniform edge costs) is NP-hard [28]. Thus, Steiner-kEC is an NP-hard problem. Since ST is a special case of SNDP, SNDP is an NP-hard problem. Thus, multi-SNDP is also an NP-hard problem. Both kEC and multi-kEC ($k \geq 2$) are also known to be NP-hard problems [15]. Finally, kVC is NP-hard [2], and thus so is VC-SNDP.

It is considered highly unlikely that efficient algorithms for exactly solving NP-hard problems exist. Thus, 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* (or performance guarantee) of the algorithm. A related concept to approximation guarantees is the concept 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 non-negative, and all not identically 0, objective cost functions, over all instances of graphs. If it can be proved that this largest ratio has value k, the linear programming relaxation is said to have a *k integrality gap*. If the integrality gap of a problem's LP relaxation is 1, then there always exists an optimal solution of its LP relaxation that is integer. The integrality gap provides a measure of the quality of the lower bound provided by Q_{LP} for Q. Furthermore, a polynomial-time constructive proof for a bound $\alpha \geq 1$ on the integrality gap provides an α-approximation algorithm for problem P. Let $opt(Q)$ be the value of an optimal solution to Q, and let $opt(Q_{LP})$ be the value of an optimal solution to Q_{LP}.

In previous work [4], we examined multi-SNDP and showed that the integrality gap for the linear programming relaxation of multi-SNDP is at most $\frac{3}{2}\frac{r_m}{r_l}$ when r_m is even, and at most $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ when r_m is odd; where r_m is the maximum edge connectivity requirement induced by the vertex connectivity requirements, and r_l is the smallest nonzero vertex connectivity requirement. We also presented an approximation algorithm for multi-SNDP which has a performance guarantee the same as this upper bound, and showed that these are both improvements for the best known bounds for certain special cases of the problem. Concerning metric 2EC, there is a known upper bound of 3/2 for the integrality gap of the LP relaxation of metric 2EC [1], and a $\frac{3}{2}$-approximation algorithm for metric 2EC [12].

Concerning VC-SNDP, there is a k-approximation algorithm for kVC on a complete graph with non-negative costs [25]; and a 2-approximation algorithm for VC-SNDP on a general graph with non-negative edge costs and vertex connectivity requirements $r_i \in \{0, 1, 2\}$ [11]. For metric 2VC, there is a $\frac{3}{2}$-approximation algorithm [12]. When edge costs are all 1 (i.e., uniform edge costs, or, equivalently, on an unweighted graph), there is a $\frac{3}{2}$-approximation algorithm for VC-SNDP with $r_i \in \{1, 2\}$ [26], and a 4/3-approximation algorithm for 2VC [30]. For a summary of the current known approximation guarantees on the different versions of kVC, see [24]. For a summary of some of the known results on SNDP and VC-SNDP, see [26].

The best known approximation guarantee for ST is an approximation guarantee of 1.39 for a weighted graph[1] [5], and a 1.28-approximation guarantee for quasi-bipartite graphs (graphs in which no two Steiner vertices are adjacent) and for complete graphs with edge costs in $\{0, 1\}$ [29]. The best known upper bound on the integrality gap of the LP relaxation of ST is 1.55 [5].

[1] A weighted graph is a graph with non-negative edge costs.

4.1.2 Overview

In this paper, we examine the special cases of both SNDP and VC-SNDP (and related variations of these problems) with low vertex connectivity requirements (i.e., $r_i \in \{0, 1, 2\}$, $i \in V$). We investigate cases where the optimal values of multi-SNDP and SNDP are equal. We demonstrate that problems multi-SNDP and metric SNDP are closely related for $r_i \in \{0, 1, 2\}$, $i \in V$. We show that the upper bound for the integrality gap of the LP relaxation of metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, is $\frac{3}{2}$, thus generalizing the known upper bound of 3/2 for the integrality gap of the LP relaxation of metric 2EC [1]. We also show that our multi-SNDP approximation algorithm [4], with a given modification, applies as well for metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, thus giving a $\frac{3}{2}$-approximation algorithm for metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$. This result extends the known result of a $\frac{3}{2}$-approximation algorithm for metric 2EC [12]. Our $\frac{3}{2}$ approximation guarantee and integrality gap bound applies to the Steiner tree problem with either uniform or metric edge costs, and improves the previous best known integrality gap bound for these cases [8].

We further show that these results for integrality gap and approximation guarantee can be expanded to include the related problem where vertex rather than edge connectivity is examined, i.e., to metric VC-SNDP $r_i \in \{0, 1, 2\}$, $i \in V$. Our $\frac{3}{2}$-approximation algorithm for metric VC-SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, is an improvement on the previously best known integrality gap bound and approximation guarantee for the problem, which was 2 [11]. It also provides a more general result than Krysta and Kumar [26], who provided a $\frac{3}{2}$-approximation algorithm for VC-SNDP with $r_i \in \{1, 2\}$, $i \in V$, on a general graph with edge costs all 1; and than Frederickson and Ja'Ja' [12], who provided a $\frac{3}{2}$-approximation algorithm for metric 2VC. Restricted to metric 2VC, our algorithm (but not the proof) is similar to the $\frac{3}{2}$-approximation algorithm for metric 2VC given by Frederickson and Ja'Ja' [12], and thus our proof can provide an alternative, more compact proof of their result.

The remainder of this chapter is organized as follows: In Section 4.2, we discuss the metric completion of a graph, and provide some of our key theorems and results from [4] to which we will be referring throughout this paper. In Section 4.3, we show how multi-SNDP and metric SNDP are closely related for $r_i \in \{0, 1, 2\}$, $i \in V$ and we investigate cases where the optimal values of multi-SNDP and SNDP are equal. In Section 4.4, we show the relationship of the integrality gaps for the LP relaxations of metric SNDP and multi-SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$; and prove that the integrality gap for the LP relaxation of metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, has an upper bound of $\frac{3}{2}$. We also show that our multi-SNDP approximation algorithm [4], with certain modifications, applies for metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, thus giving a $\frac{3}{2}$-approximation algorithm for that case. Finally, in Section 4.5, we investigate VC-SNDP, particularly for small vertex connectivity requirements. We further modify our approximation algorithm so that we get a $\frac{3}{2}$-approximation algorithm for metric VC-SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$. We also show there is a $\frac{3}{2}$ upper bound for the integrality gap of the LP relaxation of this problem.

4.2 Background

4.2.1 Metric Completion

Given a complete graph $G = (V, E)$ with non-negative edge costs c, the *metric completion*, $G^{\triangle} = (V, E)$, of G is obtained by setting the edge cost $c^{\triangle}_{ij}$ to be that of a minimum-cost path connecting vertices i and j in G, for every edge $ij \in E$. Clearly, the edge costs $c^{\triangle}$ are metric.

For any complete graph G' with metric costs, a complete graph G with non-negative edge costs can be created such that G' is the metric completion of G. Such a graph G is not necessarily unique: Let $c' \in \mathbb{R}^E$ be the edge costs of G'. Let p_{ij} be a path of length at least two (i.e., containing at least two edges) connecting vertices

i and j in G', and let $c(p_{ij})$ be the cost of that path. Let P_{ij} be the set of all such i-j paths in G'. Such a graph G is formed as follows: Take the edges ij in G' such that $c'_{ij} = \min_{p_{ij}\in P_{ij}}\{c(p_{ij})\}$, and assign them each respectively a cost c_{ij} such that $c_{ij} > \min_{p_{ij}\in P_{ij}}\{c(p_{ij})\}$. This choice is not unique, hence such a graph G is not unique. See, for example, Figure 4.1. If $c'_{ij} < \min_{p_{ij}\in P_{ij}}\{c(p_{ij})\}$ for all edges ij in G', then G will be precisely G'. In this case, such a graph G is unique. For example, take the family of complete graphs with one edge of cost 1 and the other edges of cost 2 (or edge costs consisting of any multiple of these costs). See Figure 4.2.

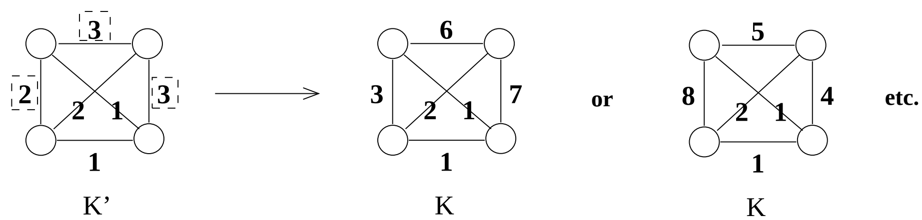

Figure 4.1: An example where such a graph K is not unique. Edge costs are shown on the edges.

Figure 4.2: The instances $|V| = 3$ and $|V| = 4$ for G', respectively, in a family of graphs for which G is precisely G'. Edge costs are shown on the edges.

On the other hand, a complete graph $G = (V, E)$ is its own metric completion when there are no edges $ij \in E$ with cost greater than the cost of a minimum-cost path connecting vertices i and j in G. In other words, a complete graph is its own metric completion when the cost of each of the edges $ij \in E$ is precisely the cost of a minimum-cost path connecting vertices i and j in G.

Proposition 1 *Every complete graph $G' = (V, E)$ with metric edge costs $c \in \mathbb{R}^E$ is its own metric completion; i.e., $G'^{\triangle} = G'$.*

Proof: Let $c^{\triangle} \in \mathbb{R}^E$ be the edge costs of the metric completion $G'^{\triangle}$. Let p_{ij} be a path of length at least two (i.e., containing at least two edges) connecting vertices i and j in G', and let $c_{p_{ij}}$ be the cost of that path. Let P_{ij} be the set of all such i-j paths in G'. Since G' has metric costs, $c_{ij} \leq \min_{p_{ij} \in P_{ij}}\{c_{p_{ij}}\}$ for each $ij \in E$. Then, $c^{\triangle}_{ij} = \min\{c_{ij}, \min_{p_{ij} \in P_{ij}}\{c_{p_{ij}}\}\} = c_{ij}$ for all $ij \in E$. Thus, $G'^{\triangle} = G'$. ■

Proposition 2 *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$. Let $G^{\triangle}$ be the metric completion of G. The following holds:*

$$opt(MST \text{ on } G) \leq opt(MST \text{ on } G^{\triangle}).$$

Proof: Let $M^{\triangle}$ be a minimum cost spanning tree of $G^{\triangle}$. Let M' be the spanning (multi-)subgraph of G obtained by replacing each edge in $M^{\triangle}$ by the corresponding minimum cost path in G that gave rise to its edge cost. Notice that the total cost of the two subgraphs are equal, i.e., $cost(M^{\triangle}) = cost(M')$. Since M' may contain cycles and/or multiple copies of edges, the cost of a spanning tree, T, of M' is less than or equal to the cost of M'. Further, since M' is a spanning multi-subgraph of G, T is also a spanning tree of G. Thus, $opt(MST \text{ on } G) \leq cost(T) \leq cost(M') = cost(M^{\triangle}) = opt(MST \text{ on } G^{\triangle})$. ■

4.2.2 Multi-SNDP, and Background Results

Given a complete graph $G = (V, E)$, and any vertex set $S \subset V$, let $\delta(S)$ denote the set of edges $\{uv \in E : u \in S, v \notin S\}$. Let $G[S]$ be the subgraph induced by the vertex set S, i.e., the subgraph consisting of the vertices S and all edges of G that have both ends in S. Problem multi-SNDP can be formulated as an integer linear program (ILP) as follows [21]:

$$
\begin{aligned}
\text{minimize} \quad & cx && (4.1)\\
\text{subject to} \quad x(\delta(S)) \;\geq\; & \max_{ij\in\delta(S)} r_{ij}, \quad \text{for all } \emptyset \subset S \subset V, && (4.2)\\
x_e \;\geq\; & 0, \quad \text{for all } e \in E, && (4.3)\\
& x_e \quad \text{integer}. && (4.4)
\end{aligned}
$$

Constraints (4.2) are called *cut constraints*, and constraints (4.3) are called *non-negativity constraints*. The linear programming (LP) relaxation of multi-SNDP is ILP(4.1) without the integer constraints (4.4), and will be referred to as LP(4.1). Notice that, in a complete graph $G = (V, E)$, $\max_{ij\in\delta(S)} r_{ij} = \min\{\max_{i\in S} r_i, \max_{j\in V\setminus S} r_j\}$, for all $\emptyset \subset S \subset V$. For example, see Figure 4.3.

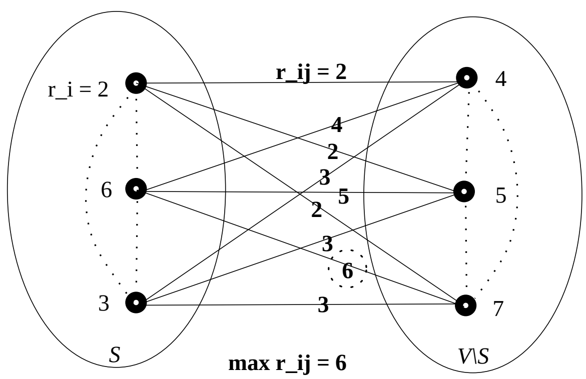

Figure 4.3: An example of $\max_{ij\in\delta(S)} r_{ij} = \min\{\max_{i\in S} r_i, \max_{j\in V\setminus S} r_j\}$ in a complete graph.

Given any multi-SNDP problem, i.e., given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$, let $r_m = \max\{r_{ij} \mid i, j \in V\}$, $r_l = \min\{r_{ij} \mid r_{ij} > 0,\ i, j \in V\}$, and $V^+ = \{i \in V \mid r_i > 0\}$. Let $G^+ = (V^+, E_{V^+})$ be the complete graph with vertex set V^+, respective vertex connectivity requirements from G, edge set E_{V^+} consisting of the edges E that have both ends in V^+, and each edge costs c^+_{ij}, $i, j \in V^+$ equal to the cost of a minimum-cost i-j path in G. Furthermore, let *opt*(*multi-SNDP*) be the cost of an optimal multi-SNDP solution (i.e., the optimal value of the integer linear program (4.1)) on G, and let *opt*(*multi-SNDP*$_{LP}$) be the cost of an optimal solution to the LP relaxation of multi-SNDP (i.e., the optimal value of the linear program LP(4.1)) on G. For $\emptyset \subset S \subset V^+$, the value $\max_{ij\in\delta(S)} r_{ij}$ is the same for G^+ and $G^\triangle$ (and for G) [4].

We list some theorems and results from our previous work [4], to which we will be referring in this paper.

Lemma 3 (Lemma 6, Boyd and Cameron [4]) *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$,*

$$opt(multi\text{-}SNDP) \ \leq \ opt(SNDP).$$

Also,

$$opt(multi\text{-}SNDP_{LP}) \ \leq \ opt(SNDP_{LP}).$$

Theorem 4 (Goemans and Bertsimas [16]) *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$ and a non-negative connectivity requirement r_{ij} for every (unordered) pair of vertices $i, j \in V$. Let $G^\triangle = (V, E)$ be the metric completion of G, with metric costs $c^\triangle \in \mathbb{R}^E$. Then the optimal values of multi-SNDP on G and on $G^\triangle$ are equal (and the optimal values of their linear programming relaxations are equal).*

Corollary 5 (Corollary 5, Boyd and Cameron [4]) *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and the metric completion $G^\triangle$ of G,*

$$opt(\text{multi-}kEC \text{ on } G) = opt(\text{multi-}kEC \text{ on } G^\triangle).$$

Using new upper bounds for the optimal values of the minimum cost T-join and the minimum cost spanning tree in terms of the optimal value of the LP relaxation of multi-SNDP, we obtained the following upper bound on the integrality gap of multi-SNDP:

Theorem 6 (Theorem 13, Boyd and Cameron [4]) *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. The integrality gap for the linear programming relaxation of multi-SNDP is at most $\frac{3}{2}\frac{r_m}{r_l}$ when r_m is even, and at most $\frac{3r_m+1}{2r_l} = \frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ when r_m is odd.*

The following is an approximation algorithm for multi-SNDP [4]:

Algorithm A

Input: A complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$.
Output: A r_m-edge connected spanning multi-subgraph (i.e., subgraph with multiple copies of the edges allowed) of G, having cost within $\frac{3}{2}\frac{r_m}{r_l}$ of the optimal solution value for multi-SNDP, if r_m is even, and within $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ of the optimal solution value for multi-SNDP, if r_m is odd.

A0. Form the complete graph G^+ with vertex set V^+, respective vertex connectivity requirements from G, and each edge cost c^+_{ij}, $i, j \in V^+$, equal to the cost of a minimum-cost i-j path in G.

A1. Find a minimum cost spanning tree, $M = (V, F)$, of G^+ and take $\lceil \frac{r_m}{2} \rceil$ copies of its edges. Let this form the edge set E'. Let $V^{odd} \subset V$ be the set of vertices of G^+ having odd degree in M.

A2. Find a minimum cost T-join $J \subseteq E$, on G^+, where $T = V^{odd}$, i.e., M has a corresponding V^{odd}-join. This is accomplished by finding a minimum cost "pairing" of the odd degree vertices of M using shortest paths from G^+. Take $\lfloor \frac{r_m}{2} \rfloor$ copies of the edges in J, and let this form the edge set J'.

A3. Combine the minimum cost spanning trees from (A1) and the T-joins from (A2), i.e., form the multi-subgraph $K^+ = (V, E' \cup J')$.

A4. Form a multi-subgraph, K, of G from K^+ by adding the Steiner vertices of G to K^+ and by replacing each edge of K^+ by the corresponding minimum-cost path in G that gave rise to its edge cost.

We now proceed to the main results of this paper.

4.3 Investigation of Equality in Lemma 3, and Further Results

Given a complete graph $G' = (V, E)$ with metric edge costs and $|V| \geq 3$, Alexander, Boyd, and Elliott-Magwood ([1], Lemma 3) show that there is always an optimal solution of multi-2EC that is simple. In particular, this means that equality holds in (the first part of) Lemma 3 for the special case where $r_i = k = 2$ for all $i \in V$ in a complete graph $G' = (V, E)$ with metric edge costs and $|V| \geq 3$.

Theorem 7 ***(Lemma 3, Alexander, Boyd, and Elliott-Magwood [1])*** *Given a complete graph $G' = (V, E)$, with $|V| \geq 3$ and with metric edge costs $c' \in \mathbb{R}^E$,*

$$opt(multi\text{-}2EC \text{ on } G') = opt(2EC \text{ on } G').$$

Problems multi-2EC and metric 2EC are closely related. This is shown by the following proposition.

Proposition 8 *Given a complete graph $G = (V, E)$, $|V| \geq 3$, with non-negative edge costs $c \in \mathbb{R}^E$, let $G^\triangle = (V, E)$ be the metric completion of G, with metric costs $c^\triangle \in \mathbb{R}^E$. Then*

$$opt(multi\text{-}2EC \text{ on } G) \;=\; opt(2EC \text{ on } G^\triangle).$$

Proof: Set $k = 2$ in Corollary 5 and apply Theorem 7 with $G' = G^\triangle$. ∎

We will show in Corollary 13 and Theorem 14, that Theorem 7 and Proposition 8 can respectively be generalized to complete graphs with vertex connectivity requirements $r_i \in \{0, 1, 2\}$ for all vertices i, and at least three vertices with vertex connectivity requirement 2, i.e. that equality holds in (the first part of) Lemma 3 for the special case of metric edge costs and $r_i \in \{0, 1, 2\}$, for all vertices i.

4.3.1 Equality Does Not Hold in General

It is important to note that equality in (the first part of) Lemma 3 does not hold in general. In fact, equality does not even hold in general for kEC and multi-kEC for $k > 2$, even for metric edge costs. For example, take the family of complete graphs on $k + 1$ vertices, $k > 2$. Arrange the vertices in a circle. Assign edge costs 1 to all the "outside" edges, and edge cost 2 to all the "inside" edges. These edge costs are metric. The optimal kEC solution is the graph itself, with total cost $1(k + 1) + 2[\frac{(k-2)(k+1)}{2}] = k^2 - 1$ (note that the only simple graph that is k-edge connected on $k + 1$ vertices is the complete graph). An optimal multi-kEC solution, for k even, is obtained by taking $\frac{k}{2}$ copies of all the outside edges, with total cost $\frac{k}{2}(k + 1) < k^2 - 1$. An optimal multi-$k$EC solution, for k odd, is obtained by taking $\frac{k+1}{2}$ copies of k of the outside edges and $(\frac{k+1}{2}) - 1$ copies of the remaining outside edge, with total cost $(\frac{k+1}{2})(k) + [(\frac{k+1}{2}) - 1] = \frac{1}{2}(k^2 + 2k - 1) < k^2 - 1$. Thus, for this family of graphs with metric edge costs and $k > 2$, $opt(\text{multi-}kEC) < opt(kEC)$. Figures 4.4 and 4.5 illustrate this family of graphs for $k = 3$ and $k = 4$, respectively.

Figure 4.6 provides another counter example that demonstrates that equality in (the first part of) Lemma 3 does not hold in general.

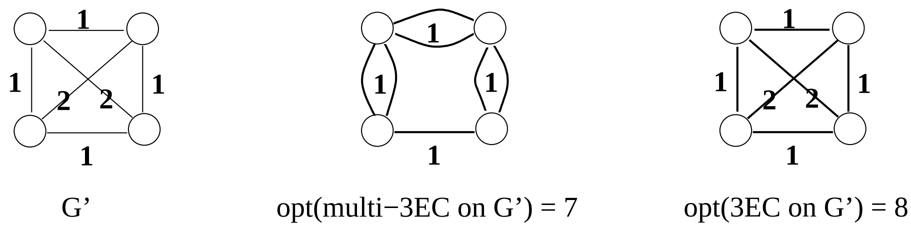

Figure 4.4: $k = 3$ in the family of graphs. Edge costs are shown on the edges.

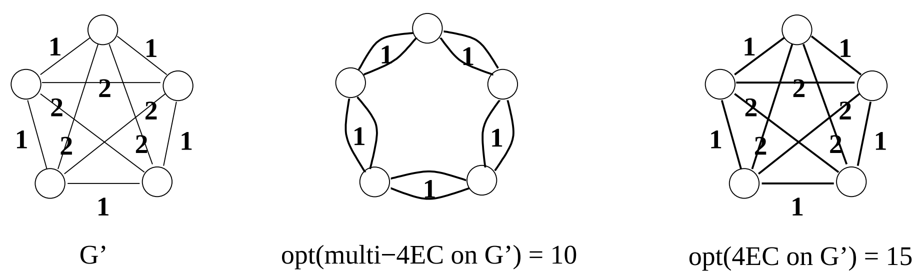

Figure 4.5: $k = 4$ in the family of graphs. Edge costs are shown on the edges.

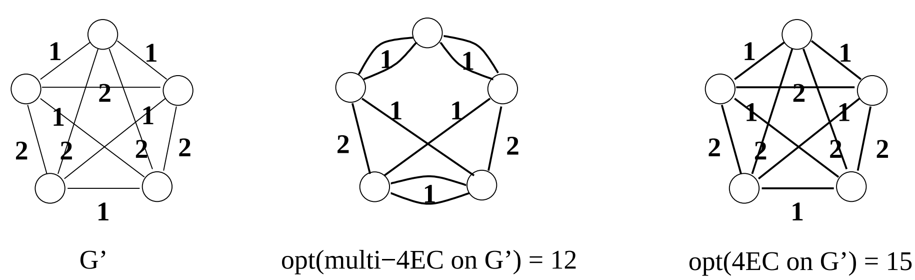

Figure 4.6: Another counter example of equality in Lemma 3. Edge costs are shown on the edges.

Further, notice that equality in (the first part of) Lemma 3 still does not hold in general for kEC and multi-kEC, $k > 2$, even when the number of vertices in a complete graph with metric edge costs is increased past a certain point so that there is a "large enough" number of vertices. The following is a counter example:

Take the family of complete graphs on n vertices, where $n \geq k+1$ and $k > 2$. Arrange the vertices in a circle. Assign edge costs 1 to all the "outside" edges, and edge cost 2 to all the "inside" edges. These edge costs are metric. In a feasible kEC solution, the degrees of all the vertices are at least k. Also, twice the number of edges in a feasible kEC solution is at least the sum of the vertex degrees. Thus, the number of edges in a feasible kEC solution is at least $\frac{1}{2}kn$. In an optimal kEC solution, n of those minimum $\frac{1}{2}kn$ edges are outside edges (each of cost 1), and the remainder of the edges are inside edges (each of cost 2). Thus, $opt(kEC) \geq n(1) + \frac{(k-2)n}{2}(2) = (k-1)n$.

An optimal multi-kEC solution, for k even, is obtained by taking $\frac{k}{2}$ copies of all the outside edges; with total cost $opt(multi\text{-}kEC) = \frac{1}{2}kn$. An optimal multi-$k$EC solution, for k odd and n even, is obtained by alternatingly taking $\frac{k-1}{2}$ copies of $\frac{n}{2}$ of the outside edges and $\frac{k+1}{2}$ copies of the other $\frac{n}{2}$ outside edges; with total cost $opt(multi\text{-}kEC) = \frac{k-1}{2}(\frac{n}{2}) + \frac{k+1}{2}(\frac{n}{2}) = \frac{1}{2}kn$. An optimal multi-$k$EC solution, for k odd and n odd, is obtained by alternatingly taking $\frac{k-1}{2}$ copies of $\frac{n-1}{2}$ of the outside edges, $\frac{k+1}{2}$ copies of the other $\frac{n}{2}$ outside edges, and $\frac{k+1}{2}$ copies of the last outside edge; with total cost $opt(multi\text{-}kEC) = \frac{k-1}{2}(\frac{n-1}{2}) + \frac{k+1}{2}(\frac{n+1}{2}) = \frac{1}{2}kn + \frac{1}{2}$. Thus, $opt(multi\text{-}kEC) = \frac{1}{2}kn + \frac{1}{2}$ for k odd and n odd; and $opt(multi\text{-}kEC) = \frac{1}{2}kn$ otherwise.

Notice that $k \geq 3$ implies that $\frac{k}{2} - 1 \geq \frac{1}{2}$. Combining this with the fact that $n \geq k+1 \geq 4$, we obtain that $(\frac{k}{2}-1)n \geq 2$. Thus, $(\frac{k}{2}-1)n > 0$ and $(\frac{k}{2}-1)n - \frac{1}{2} > 0$. But, $(\frac{k}{2}-1)n > 0$ is equivalent to $\frac{1}{2}kn - n > 0$, which is equivalent to $kn - n > \frac{1}{2}kn$, which is equivalent to $(k-1)n > \frac{1}{2}kn$. Also, $(\frac{k}{2}-1)n - \frac{1}{2} > 0$ is equivalent to $\frac{1}{2}kn - n - \frac{1}{2} > 0$, which is equivalent to $kn - n > \frac{1}{2}kn + \frac{1}{2}$, which is equivalent to $(k-1)n > \frac{1}{2}kn + \frac{1}{2}$. Hence, $\frac{1}{2}kn < (k-1)n$ and $\frac{1}{2}kn + \frac{1}{2} < (k-1)n$, respectively. Thus, for this family of graphs with metric edge costs, $k > 2$ and $n \geq k+1$, $opt(\text{multi-}kEC) < opt(kEC)$. Since this holds for any number of vertices greater than k, increasing the number of vertices still does not make equality hold in general in Lemma 3 for kEC and multi-kEC with $k > 2$.

Figure 4.7 provides an example with non-negative edge costs and vertex connectivity requirements that are not all the same where equality does not hold in general in (the first part of) Lemma 3.

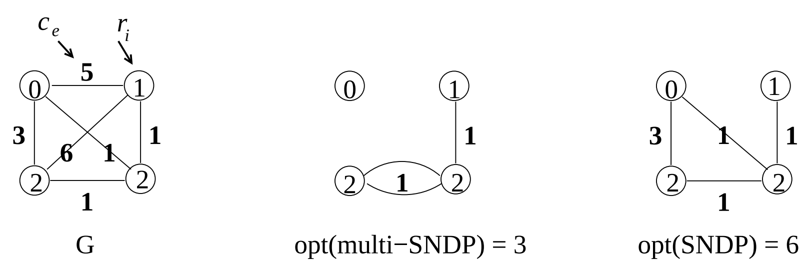

Figure 4.7: A counter example to equality in Lemma 3 that has non-uniform vertex connectivity requirements. Edge costs are shown on the edges, and vertex connectivity requirements on the vertices.

4.3.2 Equality Holds for $r_i \in \{0, 1, 2\}$

We have seen that equality holds in Lemma 3 for the special case where we are given a complete graph $G' = (V, E)$ with metric edge costs and $|V| \geq 3$, and where $r_i = k = 2$ for all $i \in V$, but does not hold in general when $r_i = k > 2$ for all $i \in V$, even for kEC and multi-kEC (for $k > 2$), even for metric edge costs; even when the number of vertices is increased past a certain point so that there is a "large enough" number of vertices. The question then arises: Does equality hold in general in Lemma 3 if $r_i \in \{0, 1, 2\}$, for all $i \in V$? We show that equality holds for $r_i \in \{0, 1\}$ for general graphs and non-negative edge costs, i.e., for ST. We also show that equality holds for $r_i \in \{0, 1, 2\}$ for complete graphs and metric edge costs, provided there are at least three $i \in V$ such that $r_i = 2$.

Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$. Let x^* be an optimal solution of multi-SNDP on G, and let x^*_e be the number of times each edge $e \in E$ is included in the solution x^*. Let H be the corresponding multi-subgraph of G, with exactly x^*_e copies of the edge e, for all $e \in E$. Without loss of generality, assume that H is edge-minimal; that is, $E(H)$ does not properly contain (as a multi-set) $E(H')$ for any optimal solution H' of multi-SNDP on G. As a point of interest, notice that any multi-edge in an optimal multi-SNDP solution M that is not edge-minimal must have edge cost 0. This is true since if M is not edge-minimal , there exists an edge (or copy of an edge) that can be removed from M without violating the cut constraints. If this edge does not have cost 0, removing it would yield a cheaper solution (which is a contradiction). Also, notice that if H is not edge-minimal, then there is another optimal multi-SNDP solution that is. This is true since if H is not edge-minimal, there exists an edge (or copy of an edge) that can be removed from H without violating the cut constraints. This edge must have cost 0, otherwise removing it would yield a cheaper solution (which is a contradiction). Removing all such extra copies of edges, we get an optimal multi-SNDP that is edge-minimal. In particular, notice that any multi-edge in an optimal multi-SNDP solution that is not edge-minimal must have edge cost 0.

Let $E(H)$ denote the set of (multi-)edges of H. Let the cut $\delta_H(S)$ in H be the set of (multi-)edges $\{uv \in E(H) : u \in S, v \notin S\}$. We call $\delta_H(S)$ a *k-cut in H* if the size of $\delta_H(S)$ is k, i.e., if $x^*(\delta_H(S)) = k$. We call a k-cut in H *tight* if there exists an $S' \subset V$, $S' \neq \emptyset$, such that $\delta_H(S')$ is the k-cut, $x(\delta(S')) \geq k$ is a constraint in (4.2), and $x^*(\delta_H(S')) = k$.

We use the following lemma repeatedly.

Lemma 9 *Let $G = (V, E)$ be a general graph with non-negative edge costs $c \in \mathbb{R}^E$, $|V| \geq 3$, and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. Let x^* and H be* *defined as above. Let* $\max_{ij \in E} r_{ij} = k$ *and let* $e' \in E$ *be such that* $x^*_{e'} \geq k$*. Then* $x^*_{e'} = k$ *and the* k *copies of* e' *form a tight* k*-cut in* H*.*

Proof: Since $\max_{ij \in E} r_{ij} = k$, the largest right hand side in the constraints (4.2) is k. Since H is edge-minimal, we thus have that $x^*_e \leq k$ for all $e \in E(H)$. Therefore, $x^*_{e'} = k$.

We claim that the k copies of e' in H are in a tight k-cut in H (i.e., there exists an $S' \subset V$, $S' \neq \emptyset$, such that $\delta_H(S')$ consists of the k copies of e', $x(\delta(S')) \geq k$ is a constraint in (4.2), and $x^*(\delta_H(S')) = k$). Suppose this is not true, i.e., suppose that the k copies of e' in H not are in a tight k-cut in H. Then for every $\hat{S} \subset V$ whose cut contains e', either the right hand side of the corresponding constraint in (4.2) is at most $k-1$; or the right hand side of the corresponding constraint in (4.2) is at most k and $x^*(\delta_H(\hat{S})) \geq k+1$. Thus, we can remove a copy of e' from H and the resulting multi-graph will still be feasible. This is a contradiction, since H is edge-minimal. Thus, the k copies of e' in H are in a tight k-cut in H. ■

Lemma 9 implies the following corollary.

Corollary 10 *Let* $G = (V, E)$ *be a general graph with non-negative edge costs* $c \in \mathbb{R}^E$*,* $|V| \geq 3$*, and vertex connectivity requirements* $r_i \in \mathbb{N}_{\geq 0}$*,* $i \in V$*. If* $\max_{ij \in E} r_{ij} = k$*, then there will never be an edge of* $k+1$ *copies in an edge-minimal optimal multi-SNDP solution on* G*.*

The following corollary applies to the Steiner tree problem.

Corollary 11 *Let* $G = (V, E)$ *be a general graph with non-negative edge costs* $c \in \mathbb{R}^E$*,* $|V| \geq 3$*, and vertex connectivity requirements* $r_i \in \{0, 1\}$*,* $i \in V$*. Then,*

$$\textit{opt}(\textit{multi-SNDP}) \;=\; \textit{opt}(\textit{SNDP}).$$

The following theorem constructively shows that if a complete graph with metric edge costs has at least three vertices with vertex connectivity requirements $\max_{ij \in E} r_{ij}$, then there exists an optimal solution to multi-SNDP which contains at most $(\max_{ij\in E} r_{ij})-1$ copies of any edge.

Theorem 12 *Let $G = (V, E)$ be a complete graph with metric edge costs $c \in \mathbb{R}^E$, $|V| \geq 3$, vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $k = \max_{ij\in E} r_{ij} \geq 2$, and with at least three vertices $i \in V$ such that $r_i = k$. Then there exists an optimal multi-SNDP solution that contains at most $k-1$ copies of any edge.*

Proof: Let H and x^* be defined as above (see the paragraph preceding Lemma 9). If H is simple, then we are done. Suppose H is not simple, i.e., suppose H has at least one multi-edge. Recall that, since H is edge-minimal, $x^*_e \leq k$ for all $e \in E$. If $x^*_e \leq k-1$ for all $e \in E$, then we are done. Suppose H has at least one multi-edge consisting of k copies of that edge. Let $e' = uv \in E$ be that edge, i.e., $x^*_{e'} = k$. From Lemma 9, there is an $S' \subset V$, $S' \neq \emptyset$, such that $\delta_H(S')$ consists of the k copies of e', $x(\delta(S')) \geq k$ is a constraint in (4.2), and $x^*(\delta_H(S')) = k$; and there exists an $i \in S'$ and an $j \in V \backslash S'$ such that $r_i = r_j = k$. Without loss of generality, assume $u \in S'$ and $v \in V \backslash S'$.

Let $H[S']$ be the subgraph of H with vertex set S' and edge set $\{fg \in E(H) \mid f, g \in S'\}$. By assumption, there exists $h \in V$, $h \neq i$, $h \neq j$, such that $r_h = k$. Without loss of generality, assume $h \in S'$. Notice that some of i, j, or h could be u or v, but at least one of them must be different from u or v. Without loss of generality, assume $i \neq u$.

Since H is an optimal solution, there exist at least k edge-disjoint paths in H between i and j. In fact, since $i \in S'$, $j \in V \backslash S'$, $\delta_H(S')$ consists of k copies of e', and $x^*_{e'} = k$, we know that $e' = uv$ is in each of these paths (and thus there are exactly k edge-disjoint i-j paths in H). Thus, in particular, there are at least k edge-disjoint paths, $P_1, P_2, \ldots, P_k$, in $H[S']$ from i to u. Let $w \in S'$ be the vertex adjacent to u in P_1. Create J from H by removing the edge uw (note that if uw is a multi-edge, then just remove one copy of it) and one copy of the edge uv, and adding the edge wv.

Since the edge costs are metric, this can be done at no increase in cost. Also note that wv was not an edge in H. Let $y \in \mathbb{R}^E$ be the solution vector corresponding to J, i.e., y_e represents the number of times each edge $e \in E$ is included in J. In other words, $y_{e'} = x^*_{e'} - 1 = k - 1$; $y_{uw} = x^*_{uw} - 1$; $y_{vw} = x^*_{vw} + 1 = 0 + 1 = 1$; and $y_e = x^*_e$ for all $e \in E \backslash \{e', uw, vw\}$. Thus, we have that $y_{e'} = k - 1$, and J has no increase in the number of copies of each multi-edge of H (a new single edge was added, but no more multi-edges or copies of multi-edges were created). See Figure 4.8. We now show that J is a feasible (multi-)SNDP solution.

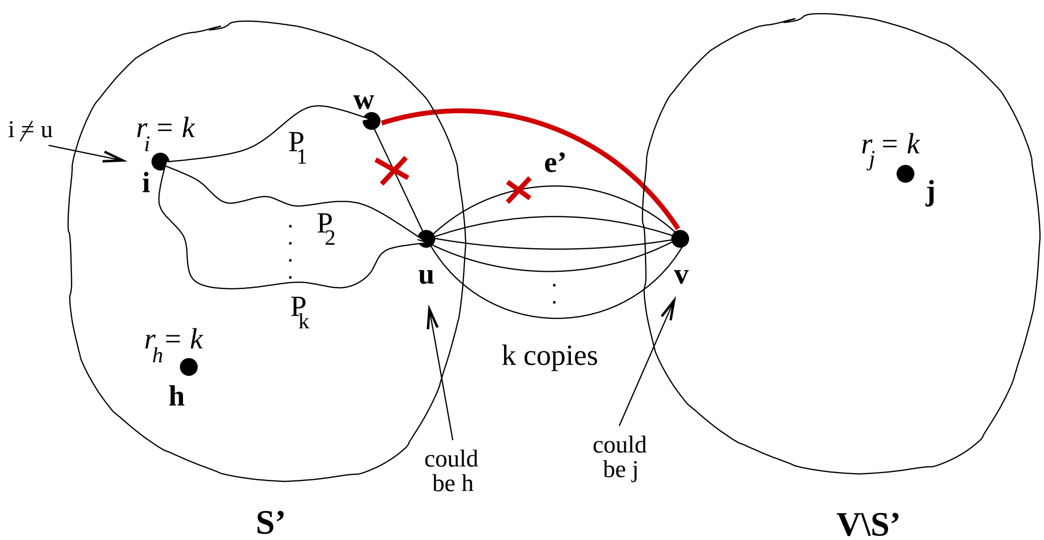

Figure 4.8: Illustration of proof of Theorem 12.

To show that J is a feasible (multi-)SNDP solution, we need to check that the cut values satisfy the cut constraints. Let $\emptyset \subset A \subset V$. We have the following cases:

(i) $uv, uw \notin \delta_H(A)$: Then $wv \notin \delta_H(A)$ and hence $\delta_H(A) = \delta_J(A)$. Thus, $y(\delta_J(A)) = x^*(\delta_H(A))$, which satisfies the constraints.

(ii) $uv \in \delta_H(A)$ and $uw \notin \delta_H(A)$: Then $vw \in \delta_J(A)$. Note that $vw \notin \delta_H(A)$, since $w \in S'$, $v \in V \backslash S'$, and $\delta_H(S')$ consists of k copies of $e' = uv$. Thus, $y(\delta_J(A)) = x^*(\delta_H(A)) - 1 + 1 = x^*(\delta_H(A))$, which satisfies the constraints.

(iii) $uv \notin \delta_H(A)$ and $uw \in \delta_H(A)$: Then $vw \in \delta_J(A)$, and the proof is the same as in (ii).

(iv) $uv, uw \in \delta_H(A)$: Then $vw \notin \delta_J(A)$. Thus, $y(\delta_J(A)) = x^*(\delta_H(A)) - 2$. Without loss of generality, assume $u \in A$. Then, $v, w \notin A$. Note also that $A \cap S' \neq \emptyset$, and $(A \cap S') \subsetneq S'$. Furthermore, notice that $\delta_H(A) = \delta_H(A \cap S') \cup \delta_H(A \cap (V \backslash S'))$. Thus, $x^*(\delta_H(A)) = x^*(\delta_H(A \cap S')) + x^*(\delta_H(A \cap (V \backslash S')))$ and $y(\delta_J(A)) = y(\delta_J(A \cap S')) + y(\delta_J(A \cap (V \backslash S')))$. But $v \notin A$ so $v \notin (A \cap (V \backslash S'))$, and thus $uv, uw \notin \delta_H(A \cap (V \backslash S'))$. By case (i), $y(\delta_J(A \cap (V \backslash S'))) = x^*(\delta_H(A \cap (V \backslash S')))$. Thus, we only need to examine the cut values of the subsets $A \cap S'$. If $\delta_J(A \cap S')$ satisfies its corresponding cut constraint, then so does $\delta_J(A)$. Let $\emptyset \subset B \subset S'$, $u \in B$, $w \notin B$. Notice that $\delta_J(B))$ contains $k-1$ copies of edge uv, and thus $y(\delta_J(B))) \geq k-1$.We have the following three cases:

(iv-a) $r_q \in \{0, 1, 2, \ldots, k-1\}$ for all $q \in B$: Recall that the cut constraint is $x(\delta(B)) \geq \max_{qd \in \delta(B)} r_{qd}$. Since $r_{qd} = \min\{r_q, r_d\} \in \{0, 1, 2, \ldots, k-1\}$ for all $qd \in \delta(B)$, the cut constraint is at most $x(\delta(B)) \geq k-1$. Notice that the k copies of uv and the (multi-)edge uw are in $\delta_H(B)$. Thus, $x^*(\delta_H(B)) \geq x^*_{e'} + x^*_{uw} \geq k+1$. Since $y(\delta_J(B)) = x^*(\delta_H(B)) - 2 \geq (k+1) - 2 = k-1$, y is a feasible solution.

(iv-b) $i \in B$: Since $i \in B \subset S'$ and $r_i = k$, and since $j \in V \backslash S'$ and $r_j = k$, then $r_{ij} = k$ and $ij \in \delta(B)$. The cut constraint is then $x(\delta(B)) \geq k$. Let P be the i-w path $P_1 - \{wu\}$. Since $i \in B$ and $w \notin B$, there exists an edge, fg, in P that is contained in the cut $\delta_H(B)$. Thus, the k copies of uv and the (multi-)edges uw, fg are in $\delta_H(B)$, and therefore $x^*(\delta_H(B)) \geq x^*_{e'} + x^*_{uw} + x^*_{fg} \geq k+2$. Since $y(\delta_J(B)) = x^*(\delta_H(B)) - 2 \geq k$, y is a feasible solution.

(iv-c) $h \in B$ and $i \in S' \backslash B$: Since $h \in B \subset S'$ and $r_h = k$, and since $j \in V \backslash S'$ and $r_j = k$, then $r_{hj} = k$ and $hj \in \delta(B)$. The cut constraint is then $x(\delta(B)) \geq k$. Since $i \in S' \backslash B$ and $u \in B$, there exists an edge, fg, in the i-u path P_2 that is contained in the cut $\delta_H(B)$. Thus, the k copies of uv and the (multi-)edges uw, fg are in $\delta_H(B)$, and therefore $x^*(\delta_H(B)) \geq x^*_{e'} + x^*_{uw} + x^*_{fg} \geq k+2$. Since $y(\delta_J(B)) = x^*(\delta_H(B)) - 2 \geq k$,

y is a feasible solution.

Therefore, y is a feasible (multi-)SNDP solution, with cost no more than the cost of x^*, with no increase in the number of copies of each multi-edge in H, with no creation of new multi-edges, and with one less copy of the multi-edge e'. Since x^* is an optimal multi-SNDP solution, the cost of y is equal to the cost of x^*. Thus, y is an optimal (multi-)SNDP solution, with no increase in the number of copies of each multi-edge in H, with no creation of new multi-edges, and with one less copy of the multi-edge e'. The above procedure can be repeated until all the k-multi-edges are removed, thus obtaining an optimal (multi-)SNDP solution that has at most $k-1$ copies of each of its multi-edges. ■

Notice that Theorem 12 does not specifically use the fact that there are at least three vertices $i \in V$ such that $r_i = k$. Rather, it uses the fact coming from that, that at least one of the vertices with connectivity type k is not an end of the k-multi-edge.

Notice that $k = \max_{ij \in E} r_{ij}$ in Theorem 12 means that either there exist at least two vertices with vertex connectivity requirement k; or that there exists at least one vertex with vertex connectivity requirement k and exactly one vertex with vertex connectivity requirement $k+1$. Theorem 12 does not hold in general either (i) when there are exactly two vertices with vertex connectivity requirement k, or (ii) when there is exactly one vertex with vertex connectivity k and exactly one vertex with vertex connectivity requirement $k+1$. Figure 4.9 provides such a counterexample. Notice that in the case of Figure 4.9, the two ends of the k-multi-edge are precisely the two vertices with vertex connectivity requirement k. Notice as well that the counterexample still holds if one of those ends has vertex connectivity requirement $k+1$. However, if $uv \in E(H)$ is a multi-edge with $x^*_{uv} = k$, and (i) holds and either $r_u \neq k$ or $r_v \neq k$ (or, (ii) holds and either $r_u \neq k$ or $r_v \neq k+1$), then the proof of Theorem 12 can be applied, and an optimal multi-SNDP solution that has $k-1$ copies of uv can be obtained. Thus, Theorem 12 is also true when there are exactly two vertices $u, v \in V$ such that $r_u = r_v = k$ (or, there is exactly one vertex $u \in V$

such that $r_u = k$ and exactly one vertex $v \in V$ such that $r_v = k+1$), and no optimal multi-SNDP solution contains k copies of the edge uv.

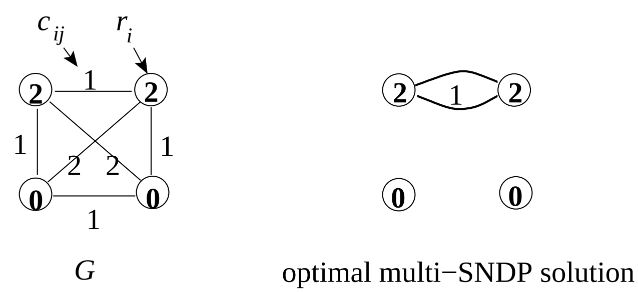

Figure 4.9: A counterexample of Theorem 12 when there are only two vertices $i \in V$ such that $r_i = k = \max_{ij \in E} r_{ij}$.

The following corollary, obtained directly from Theorem 12, generalizes Theorem 7.

Corollary 13 *Let $G = (V, E)$ be a complete graph with metric edge costs $c \in \mathbb{R}^E$, $|V| \geq 3$, vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$, and at least three vertices $i \in V$ such that $r_i = 2$. Then*

$$opt(multi\text{-}SNDP) = opt(SNDP).$$

Proof: From the proof of Theorem 12 with $k = 2$, we obtain that y is an optimal (multi-)SNDP solution with one less multi-edge. Repeating the procedure until all multi-edges are removed, we obtain an optimal SNDP solution with cost equal to that of an optimal multi-SNDP solution. ■

As with Theorem 12, Corollary 13 does not specifically use the fact that there are at least three vertices $i \in V$ such that $r_i = 2$, but rather uses the fact (coming from that) that at least one of the vertices with connectivity type 2 is not an end of the multi-edge. As we show in the next section, Corollary 13 does not hold in general when G has exactly two vertices $i, j \in V$ such that $r_i = r_j = 2$ and an optimal multi-SNDP solution of G has ij as a multi-edge. We will see in Section 4.3.3 that Corollary 13 does not hold in general when there are only two vertices that have a vertex connectivity requirement of 2.

In the proof of Theorem 12, notice that for tight k-cuts, $k = \max_{ij \in E} r_{ij}$, $k \geq 3$, that involve a single multi-edge e', once we shortcut and remove one of the copies of e', we no longer have a tight cut consisting of just multiple copies of e'. Repeating this on all of the tight k-cuts consisting of single multi-edges, we obtain an optimal multi-SNDP solution whose tight k-cuts all contain at least one other (multi-)edge in addition to a multi-edge. Notice that this never happens for $r_i \in \{0, 1, 2\}$, and is what the proof of Corollary 13 relies upon. Thus, the proof of Corollary 13 does not hold for $\max_{ij \in E} r_{ij} \geq 3$.

The following theorem shows that we can partially generalize our result in Corollary 13 to complete graphs with general edge costs, and that the problems multi-SNDP on G and SNDP on the metric completion of G are closely related for G with vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$, and at least three vertices $i \in V$ such that $r_i = 2$. The following theorem also generalizes Proposition 8.

Theorem 14 *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, $|V| \geq 3$, vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$, and at least three vertices $i \in V$ such that $r_i = 2$. Let $G^{\triangle} = (V, E)$ be the metric completion of G, with metric edge costs $c^{\triangle} \in \mathbb{R}^E$. Then*

$$opt(\textit{multi-SNDP on } G) \;=\; opt(\textit{SNDP on } G^{\triangle}).$$

Proof: Follows directly from Corollary 13 and Theorem 4. ■

Thus, equality in (the first part of) Lemma 3 does not hold in general when $r_i = k > 2$ for all $i \in V$, nor does it hold in general for kEC and multi-kEC (for $k > 2$), even when the edge costs are metric. However, equality does hold for general graphs with non-negative edge costs and $r_i \in \{0, 1\}$ (i.e., for ST); as well as for complete graphs with metric edge costs and $r_i \in \{0, 1, 2\}$ (provided there are at least three $i \in V$ such that $r_i = 2$).

4.3.3 Special Case

Notice that Corollary 13 does not hold in general when there are only two vertices $i \in V$ such that $r_i = 2$. This is because this case includes the special case when G has exactly two vertices $i, j \in V$ such that $r_i = r_j = 2$ and an optimal multi-SNDP solution of G has ij as a multi-edge. Figure 4.10 is a counterexample that illustrates why Corollary 13 does not hold in general for this special case when $r_i \in \{1, 2\}$, and Figure 4.12 is a counterexample that illustrates why it does not hold in general for this special case when $r_i \in \{0, 2\}$.

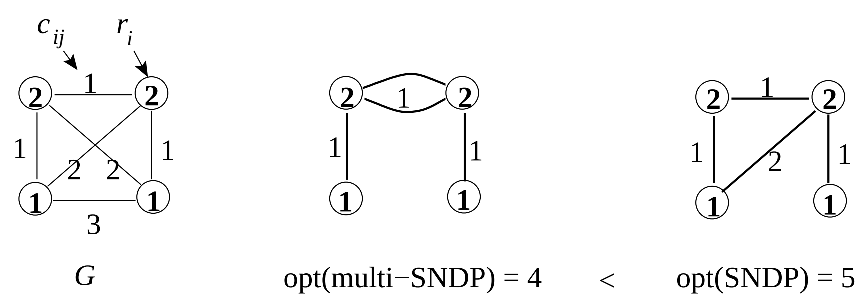

Figure 4.10: A counterexample of Corollary 13 for the special case. Edge costs are shown on the edges, and vertex connectivity requirements on the vertices. Edge costs are shown on the edges, and vertex connectivity requirements on the vertices.

Notice that, in general, the "normal" shortcutting can not be done in the special case: Let H, x^*, and S' be defined as before. Let $u, v \in V$ be such that $r_u = r_v = 2$, and let $r_i = 1$ for all $i \in V \backslash \{u, v\}$. Let $e' = uv$, $x^*_{e'} = 2$, and $x^*_e \in \{0, 1\}$ for all $e \in E \backslash \{e'\}$. Without loss of generality, assume $u \in S'$ and $v \in V \backslash S'$. Since $|V| \geq 3$, there exists at least one vertex in G other than u and v. Without loss of generality, assume $|S'| \geq 2$. From the cut constraints, $x^*(\delta_H(V \backslash S' \cup \{u\})) \geq 1$. Thus, there exists $k \in S'$, $k \neq u$, such that $uk \in \delta_H(V \backslash S' \cup \{u\})$. Note that $r_k = 1$. Create J (respectively, y) from H (respectively, x^*) by removing edges ku, uv and adding the edge kv. Since the edge costs are metric, this can be done at no increase in cost. Let $B \subset S'$, $B \neq S'$, such that $u \in B$. Since $uv \in \delta(B)$, the cut constraint for B is $x(\delta(B)) \geq 2$. But $y(\delta_J(B)) \geq 1$, thus y is not a feasible solution. Therefore, the method of shortcutting can not be done in the special case.

Notice that this does not mean that an optimal SNDP solution of equal cost to an optimal multi-SNDP solution does not exist; it only means that such a solution can not be obtained from shortcutting the multi-SNDP solution as in Theorem 12 (in this special case). As an example, see Figure 4.11.

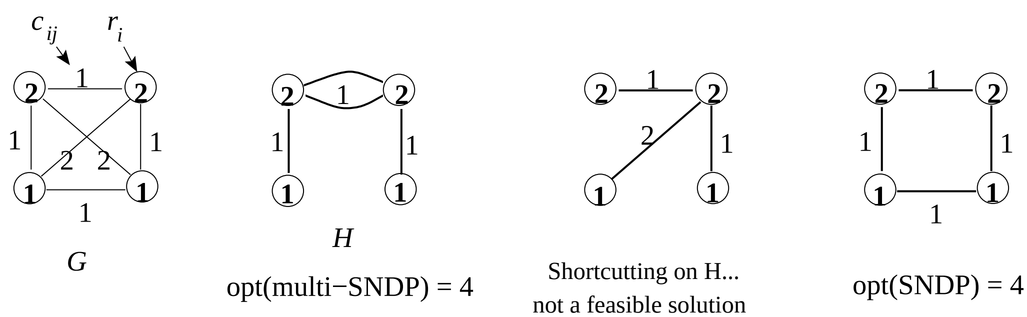

Figure 4.11: An instance of the special case, in which an optimal SNDP solution exists at equal cost to an optimal multi-SNDP solution, but it can not be obtained from shortcutting the multi-SNDP solution.

The following is a counterexample for why equality does not hold in general for this special case when $r_i \in \{0, 2\}$.

Let $G = (V, E)$ be the complete graph in Figure 4.12, with metric edge costs as shown. Let i, j be the two vertices in V with $r_i = r_j = 2$. Thus, $r_{ij} = 2$ and $r_{gf} = 0$ for all edges $gf \in E$, $gf \neq ij$. It is easily seen that all the cuts $\delta(S)$ must have value at least 0, except for the cuts corresponding to the subsets $S = \{i\}$ such that $r_i = 2$, or $S = \{i, k\}$ such that $r_i = 2$ and $r_k = 0$. The optimal multi-SNDP

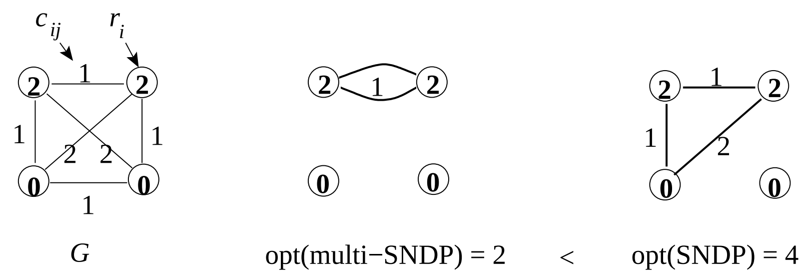

Figure 4.12: A counterexample of equality in general in Lemma 3 for $r_i \in \{0, 2\}$.

solution is obtained by taking two copies of the edge ij, for a total cost of 2. To obtain a feasible SNDP solution, the cut constraints $x(\delta(\{i\})) \geq 2$ and $x(\delta(\{j\})) \geq 2$ imply that the solution must contain at least three edges. Since the edge costs are all at least one, a feasible solution for SNDP must have cost at least three. Thus, an optimal SNDP solution for this example will have cost strictly greater than that of an optimal multi-SNDP solution.

4.3.4 Other Properties of Multi-SNDP for Small r_i's

As defined earlier, given a complete graph $G = (V, E)$ with metric edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \{0, 1, 2\}$, let x^* be an optimal solution of multi-SNDP on G. Let H be the corresponding multi-subgraph of G, with exactly x^*_e copies of the edge e, for all $e \in E$. Suppose H has at least one multi-edge. Let $e' = uv \in E$ be that edge, where $u \in S'$ and $v \in V \backslash S'$. Then, from Lemma 9, we have that $x^*_{e'} = 2$ and e' is in a tight 2-cut in H (i.e., there is an $S' \subset V$, $S' \neq \emptyset$, such that $x(\delta(S')) \geq k$ and $x^*(\delta_H(S')) = k$). In addition to Lemma 9, there are a few other properties regarding parallel edges and tight cuts in solutions for multi-SNDP when all the vertex connectivity requirements are small. We see this in the following lemmas.

Lemma 15 *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, $|V| \geq 3$, and vertex connectivity requirements $r_i = \{1, 2\}$. If $e' = uv \in E$ is a multi-edge of an optimal solution H of multi-SNDP, $S' \subset V$, $u \in S'$, $v \in V \backslash S'$, and $\delta_H(S')$ is a tight cut, then e' is not in any tight 2-cut other than $\delta_H(S')$.*

Proof: Suppose not, i.e., assume there is an $\emptyset \subset A \subset S'$, $A \neq S'$, such that the cut constraint is $x(\delta(A)) \geq 2$, and $\delta_H(A)$ consists of the 2 copies of e'. Hence, $x^*(\delta_H(A)) = 2$. Since $e' = uv \in \delta_H(A)$, without loss of generality assume $u \in A$. Consider the subset $D = S' \backslash A$. Since $V = D \cup (S' \cap A) \cup (V \backslash S')$, then $\delta_H(D) = \delta_{H[S']}(D) \cup \delta_{H[V\backslash A]}(D)$. Since $u \notin D$ and $D \subset S'$, there are no edges in $\delta_H(D)$ between D and $V \backslash S'$. Also, since $u \in A$, $v \in V \backslash S'$, and $\delta_H(A)$ consists of the 2 copies of uv, there are no edges in $\delta_H(D)$ between D and $S' \cap A$. Thus, $\delta_H(D) = \emptyset$. Thus, $x^*(\delta_H(D)) = 0$. This is a contradiction, since the cut constraint is at least $x(\delta_H(D)) \geq 1$ and x^* is an optimal solution. Thus, e' is not in any tight 2-cut other than $\delta_H(S')$. ■

Lemma 16 *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, $|V| \geq 3$, and vertex connectivity requirements $r_i = \{0, 1, 2\}$. Let $e' = uv \in E$ be a multi-edge of an optimal solution H of multi-SNDP, $S' \subset V$, $u \in S'$, $v \in V \backslash S'$, and $\delta_H(S')$ be a tight cut. If e' is in any tight 2-cut other than $\delta_H(S')$, say $\delta_H(A)$, $\emptyset \subset A \subset S$, $A \neq S'$, where $u \in A$, $v \notin A$, then $r_i = 0$ for all $i \in S' \backslash A$.*

Proof: Let $D = S' \backslash A$. From the proof of Lemma 15, $x^*(\delta_H(D)) = 0$. Thus, either $r_i = 0$ for all $i \in D$, or $r_i = 0$ for all $i \in V \backslash D$. Since there exists $j \in V \backslash S'$ such that $r_j = 2$, and $V \backslash S' \subset V \backslash D$, therefore we must have that $r_i = 0$ for all $i \in D$. ■

Lemma 17 *Let $G = (V, E)$ be a complete graph with non-negative edge costs $c \in \mathbb{R}^E$, $|V| \geq 3$, and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$. Let $e \in E$ be an edge such that $x^*_e = 1$ and e is in a tight 1-cut in H. Then e is not also in a tight 2-cut in H.*

Proof: Suppose not, i.e., assume there is an $\emptyset \subset A \subset S$ such that the cut constraint is $x(\delta(A)) \geq 2$, and such that $e \in \delta_H(A)$ and $x^*(\delta_H(A)) = 2$. Let $\delta_H(B)$, $\emptyset \subset B \subset V$, be the tight 1-cut in H containing e. So $x^*(\delta_H(B)) = 1$ and $\delta_H(B) = \{e\}$. This means that either $r_i \in \{0,1\}$ for all $i \in V \backslash B$ and there is at least one $i \in V \backslash B$ such that $r_i = 1$; or, $r_i \in \{0,1\}$ for all $i \in B$ and there is at least one $i \in B$ such that $r_i = 1$. Without loss of generality, assume that $r_i \in \{0,1\}$ for all $i \in V \backslash B$ and there is at least one $q \in V \backslash B$ such that $r_q = 1$. Consider the subset $D = B \backslash A$. Then there exists at least two vertices $i, j \in V$ such that $r_i = r_j = 2$. Since $r_i \in \{0,1\}$ for all $i \in V \backslash B$, there exists $i, j \in B$ such that $r_i = r_j = 2$. Furthermore, since the cut constraint is $x(\delta(A)) \geq 2$, without loss of generality assume $i \in A \cap B$ and $j \in B \backslash A = D$. The cut constraint corresponding to D is $x(\delta(D)) \geq 2$. There are two cases:

Case (a): If $e \notin \delta_H(D)$, then the only edges in $\delta_H(D)$ are between D and $A \cap B$. Thus, there must exist at least two edges, f, g, in H between D and $A \cap B$. Therefore, $\{e, f, g\} \in \delta_H(A)$ and thus $x^*(\delta_H(A)) \geq 3$, which is a contradiction.

Case (b): If $e \in \delta_H(D)$, then, since $x(\delta(D)) \geq 2$, there must exist at least one edge in H between D and $A \cap B$. Since $x^*(\delta_H(A)) = 2$, there can in fact be only one edge in H between D and $A \cap B$. Furthermore, this can be the only edge in the cut $\delta_H(A \cap B)$ (since $e \in \delta_H(D)$ and $\delta_H(B) = \{e\}$). Thus, $x^*(\delta_H(A \cap B)) = 1$. However, the cut constraint corresponding to $A \cap B$ is $x(\delta(A \cap B)) \geq 2$ and x^* is an optimal solution. Hence, this is a contradiction.

Therefore, if an edge e such that $x^*_e = 1$ is in a tight 1-cut in H, then e will not also be in a tight 2-cut in H. ■

4.4 Metric SNDP for $r_i \in \{0, 1, 2\}$

In 2006, Alexander, Boyd, and Elliott-Magwood showed that the integrality gap of the LP relaxation of multi-2EC (and metric 2EC) is $\leq 3/2$ [1], and gave the exact integrality gap for graphs of up to 10 vertices, and a tight lower bound for the integrality gap for graphs of 11 to 14 vertices. In both [6] and [1], it is mentioned that examples exist with an integrality gap ratio which asymptotically approaches 6/5, thus indicating that the integrality gap for the LP relaxation of multi-2EC (and metric 2EC) is between 6/5 and 3/2. In 1998, Carr and Ravi [6] conjectured that the integrality gap is in fact 4/3, and presented a result that provides some support for this. In 1982, Frederickson and Ja'Ja' presented a 3/2-approximation algorithm that applies to metric 2EC [12].

We now show some results on the upper bound of the integrality gap for the linear programming relaxation of metric SNDP with $r_i \in \{0, 1, 2\}$ (in particular, that there is an upper bound of 3/2 for its integrality gap), as well as a 3/2-approximation algorithm for metric SNDP with $r_i \in \{0, 1, 2\}$. This extends the above-mentioned results of [1] and [12], respectively.

The following lemma gives the relationship between the integrality gaps for the linear programming relaxation of multi-SNDP and for the linear programming relaxation of metric SNDP for $r_i \in \{0, 1, 2\}$; in particular, the relationship between the linear programming relaxation of multi-2EC and the linear programming relaxation of metric 2EC.

Lemma 18 *Let $G' = (V, E)$ be a complete graph with metric edge costs $c' \in \mathbb{R}^E$, $|V| \geq 3$, vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$, and at least three vertices $i \in V$ such that $r_i = 2$. Then the integrality gap of the LP relaxation of SNDP on G' is less than or equal to the integrality gap of the LP relaxation of multi-SNDP on G'. (In particular, the integrality gap of the LP relaxation of 2EC on G' is less than or equal to the integrality gap of the LP relaxation of multi-2EC on G'.)*

Proof: By Lemma 3, $opt(multi\text{-}SNDP_{LP}\ on\ G') \leq opt(SNDP_{LP}\ on\ G')$. By Corollary 13, $opt(\text{multi-}SNDP\ on\ G') = opt(SNDP\ on\ G')$. We therefore have that

$$\frac{opt(SNDP\ on\ G')}{opt(SNDP_{LP}\ on\ G')} \leq \frac{opt(multi\text{-}SNDP\ on\ G')}{opt(multi\text{-}SNDP_{LP}\ on\ G')}.$$

■

The following lemma is obtained by combining Lemma 18 and Theorem 6. It both agrees with, and generalizes, Alexander, Boyd, and Elliott-Magwood's result that the integrality gap of the LP relaxation of multi-2EC (and metric 2EC) is $\leq 3/2$ [1].

Lemma 19 *Given a complete graph $G' = (V, E)$, $|V| \geq 3$, with metric edge costs $c' \in \mathbb{R}^E$, vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$, and at least three vertices $i \in V$ such that $r_i = 2$, the integrality gap of the LP relaxation of metric SNDP is at most $\frac{3}{2}$. (In particular, the integrality gap of the LP relaxation of metric 2EC is at most $\frac{3}{2}$.)*

Proof: By Lemma 18 and Theorem 6, the integrality gap of the LP relaxation of metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, and at least three vertices $i \in V$ such that $r_i = 2$, is at most $\frac{3}{2}\frac{r_m}{r_l}$. Here, $r_m = 2$. The constructive proof of Theorem 6 returns a r_m-edge connected spanning multi-subgraph of G'. Thus, without loss of generality, set $r_l = 2$. The result follows. ■

4.4.1 Approximation Algorithm for Metric SNDP with $r_i \in \{0, 1, 2\}$

In 1982, Frederickson and Ja'Ja' examined the biconnected augmentation problem on complete graphs with metric edge costs and $F = \emptyset$ [12], i.e., metric 2VC. Following a similar idea to that of Christofides' $\frac{3}{2}$-approximation algorithm for the metric travelling salesman problem [8], Frederickson and Ja'Ja' presented a $\frac{3}{2}$-approximation algorithm for metric 2EC and metric 2VC.

Given a complete graph $G' = (V, E)$, $|V| \geq 3$, with metric edge costs, the approximation algorithm, Algorithm A, for multi-kEC that we presented in [4] (and which we re-stated for reference in Section 4.2.2 of this paper) can be modified to get a $\frac{3}{2}$-approximation algorithm for metric 2EC that is similar to that of Frederickson and Ja'Ja''s. This modification can be extended even further to get a $\frac{3}{2}$-approximation algorithm for metric SNDP for $r_i \in \{0, 1, 2\}$, $i \in V$, $\max_{ij \in E} r_{ij} = 2$. The modification is as follows: Set $r_m = 2$ and let K be the heuristic solution obtained from our approximation algorithm (cf [4], Algorithm A) applied to G'. If just metric 2EC is being considered, then $r_l = 2$ and steps (A0) and (A4) may be omitted, since there are no Steiner vertices. Note that K is a 2-edge connected spanning multi-subgraph. Also note that any multi-edges of K consist of exactly two parallel edges. (Otherwise, we can remove at least one copy of the parallel edges without affecting the 2-edge connectivity, thus obtaining a cheaper solution.) We can remove the parallel edges of K as follows [12]:

A5. For each edge uv in $J' \cap E'$: (a) remove uv from J'; (b) find an adjacent edge ux or vw in E', remove it from E' and replace it with either vx or uw, respectively; (c) insert the remaining edges of J' into E', and call this combined set of edges $\tilde{E}$. Form the resulting subgraph $H = (V, \tilde{E})$.[2]

[2] Alternatively, we can remove the parallel edges of K as follows: For each parallel edge uv in

Theorem 20 *Given a complete graph $G' = (V, E)$, $|V| \geq 3$, with metric edge costs $c' \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$, the approximation algorithm Algorithm A [4] (with $r_m = r_l = 2$), with the addition of step (A5) above, is a $\frac{3}{2}$-approximation algorithm for metric 2EC (for this case, steps (A0) and (A4) may be omitted), and for metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, respectively.*

Proof: Set $r_m = r_l = 2$ and let $\hat{H}$ be the heuristic solution obtained from our approximation algorithm ([4], Algorithm A) applied to G'. By [4], $\hat{H}$ is a feasible multi-2EC solution. Let H be the resulting subgraph after applying the transformation in the above step (A5) to $\hat{H}$.

Note that, by construction, all the vertices of $\hat{H}$ have even degree. Without loss of generality, assume that the edge ux is removed along with uv in the step (A5) above, and xv is added. The degree of u decreases by two, and the degree of x and v remain the same. Since the vertex u was incident with the parallel edge uv and the edge ux, and its degree was even, therefore the degree of u was greater than or equal to four. Thus, decreasing the degree of u by two does not disconnect u from the graph. Hence, the resulting subgraph H is connected and all vertices of H have even degree. As well, no new parallel edges are created, since the edge xv is not in $\hat{H}$ (if xv were in E' then (V, E') would contain a cycle (cycle $xuvx$) and thus would not be a minimum spanning tree; and if xv were in J' then J' would contain a cycle and thus not be a minimum cost T-join for $T = V^{odd}$). Since xv replaces the edge ux in edge set E', $(E' \backslash \{ux\}) \cup xv$ is the edge set of a spanning tree B in G. In addition, the edge set $J' \backslash uv$ is the edge set of a T-join with respect to the odd-degree vertices in B. Thus, we can repeat step (A5) as many times as necessary. Repeating for each

K, there exists a cut $\delta(S)$, $\emptyset \subset S \subset V$, such that $\delta(S) = \{uv\}$. Since $|V| \geq 3$, either u or v has a neighbour other than v or u, respectively. Without loss of generality, assume that u has a neighbour $w \neq v$. Deleting one copy each of uv and uw, and adding the edge vw maintains 2-edge connectivity at no increase in cost, while, at the same time, removing a parallel edge from the subgraph. This is shown by Alexander, Boyd, and Elliott-Magwood [1] in the proof of their Lemma 3.

of the parallel edges of $\hat{H}$, we see that the resulting subgraph H is connected and all vertices of H have even degree. Hence, H is 2-edge connected. H is also simple.

Due to metric costs, the transformation of step (A5) is performed at no increase in total edge cost. Thus, H is a 2-edge connected spanning simple subgraph of G' with cost at most $\frac{3}{2}opt(metric\ 2EC)$. Since step (A5) can also be done in polynomial time, this yields a $\frac{3}{2}$-approximation algorithm for metric 2EC. Thus, this also yields a $\frac{3}{2}$-approximation algorithm for metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$. ■

It is worthwhile noting that, given a complete graph $G = (V, E)$ with non-negative edge costs, a $\frac{3}{2}$-approximation algorithm for multi-2EC can be obtained by taking the metric completion $G^{\triangle}$ of G, applying Frederickson and Ja'Ja"s $\frac{3}{2}$-approximation algorithm for metric 2EC [12], and then using the transformation by Goemans and Bertsimas in the proof of their Theorem 3 [16], to transform the heuristic solution on $G^{\triangle}$ back to a heuristic solution on G at no increase in cost and no decrease in edge connectivity.

For unweighted (equivalently, edge costs all 1) SNDP on a general graph with vertex connectivity requirements $r_i \in \{1, 2\}$, $i \in V$, there is a $\frac{3}{2}$-approximation algorithm [26]. For the special case where the unweighted problem is on a complete graph, our result of Theorem 20 is a stronger, more general, result. Notice that unless the unweighted problem is on a complete graph, we cannot compare results, even if we make the general unweighted graph a complete graph by adding in the "missing" edges with appropriately large edge costs, since the edge costs would no longer be metric (and hence only our approximation algorithm for multi-SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, would apply).

4.5 VC-SNDP and kVC

Given a complete graph $G = (V, E)$ and $U \subset V$, let $G - U$ be the subgraph of G with vertex set $V \backslash U$ and edge set $\gamma_G(V \backslash U)$. Problem kVC on G can be formulated as an integer linear program (ILP) as follows [17]:

$$
\begin{aligned}
&\text{minimize} \quad cx && && \qquad (4.5)\\
\text{subject to} \quad & x(\delta_G(S)) \geq k, && \text{for all } \emptyset \subset S \subset V, && \qquad (4.6)\\
& x(\delta_{G-U}(S)) \geq 1, && \text{for all } \emptyset \subset U \subset V \text{ such that } |U| = k-1, && \\
& && \text{and for all } \emptyset \subset S \subset V \backslash U, && \qquad (4.7)\\
& 0 \leq x_e \leq 1, && \text{for all } e \in E, && \qquad (4.8)\\
& x_e && \text{integer.} &&
\end{aligned}
$$

More generally, the following is an integer linear programming formulation for VC-SNDP [19]:

$$
\begin{aligned}
&\text{minimize} \quad cx && && \qquad (4.9)\\
\text{subject to} \quad & x(\delta_G(S)) \geq \max_{ij \in \delta_G(S)} r_{ij}, && \text{for all } \emptyset \subset S \subset V, && \qquad (4.10)\\
& x(\delta_{G-U}(S)) \geq 1, && \text{for all } i, j \in V,\ i \neq j, && \\
& && \text{and for all } \emptyset \subset U \subset V \backslash \{i, j\} \text{ such that } |U| = r_{ij} - 1, && \\
& && \text{and for all } \emptyset \subset S \subset V \backslash U \text{ such that} && \\
& && \quad i \in S,\ j \notin S, && \qquad (4.11)\\
& 0 \leq x_e \leq 1, && \text{for all } e \in E, && \qquad (4.12)\\
& x_e && \text{integer.} &&
\end{aligned}
$$

Constraints (4.10) are called *(edge) cut constraints*, constraints (4.11) *node cut constraints*, and constraints (4.12) *non-negativity constraints*. Constraints (4.11) ensure that, for any pair of vertices $i, j \in V$, the removal of any $r_{ij} - 1$ vertices does not disconnect i and j (i.e., it still leaves at least one path from i to j in G). Notice that the following constraints may be given in place of constraints (4.11) [18]:

$$\begin{aligned} x(\delta_{G-U}(S)) \quad \geq \quad & \max_{ij\in\delta(S)} r_{ij} - |U|, \quad \text{for all } i,j \in V,\ i \neq j, \\ & \text{and for all } \emptyset \subset U \subset V\backslash\{i,j\} \text{ such that } |U| \leq r_{ij} - 1, \\ & \text{and for all } \emptyset \subset S \subset V\backslash U \text{ such that } i \in S,\ j \notin S. \qquad (4.13) \end{aligned}$$

These constraints ensure that, for all distinct pairs $i, j \in V$, deleting any set of vertices U such that $|U| \leq r_{ij} - 1$ leaves at least $(\max_{ij\in\delta(S)} r_{ij}) - |U|$ internally vertex-disjoint paths between i and j in the remaining graph. For all distinct pairs $i, j \in V$, when $|U| = r_{ij} - 1$, we obtain the constraints (4.11). Since the removal of any set of vertices which contains a cut set also disconnects a graph, constraints (4.13) contain redundant constraints, and constraints (4.11) defines the same solution set as the constraints (4.13).

Let a *partition* of V be a set $V_1, V_2, \ldots, V_p$ such that $\emptyset \subset V_a \subset V$, $V_a \cap V_b = \emptyset$ for all $a, b \in \{1, 2, \ldots, p\}$, and $V_1 \cup V_2 \cup \ldots \cup V_p = V$. ILP(4.5) can be strengthened by replacing the node cut constraints (4.7) by the following *node partition constraints* [17, 18]: $\frac{1}{2}\sum_{a=1}^{p} x(\delta_{G-U}(S_a)) \geq p - 1$, for all $\emptyset \subset U \subset V$ such that $|U| = k - 1$, and for all non-trivial partitions $S_1, S_2, \ldots, S_p$ of $V\backslash U$, for $p \geq 2$.

Notice that ILP (4.9) reduces to the ILPs for kVC and 2VC when $r_i = k$, or $r_i = 2$, respectively, for all $i \in V$. Also, removing the node cut constraints from the ILPs of kVC and VC-SNDP yields the ILPs for kEC and SNDP, respectively. Clearly, VC-SNDP is a special case of SNDP.

Notice that Steiner tree is the special case of both SNDP and VC-SNDP, where $r_i \in \{0, 1\}$ for all $i \in V$. Also, MST is the special case of both SNDP and VC-SNDP, where $r_i = 1$ for all $i \in V$. Thus, for the case where $r_i \in \{0, 1\}$ for all $i \in V$, or for the case where $r_i = 1$ for all $i \in V$, the problems SNDP and VC-SNDP are the same [19]. Indeed, when $r_i \in \{0, 1\}$, constraints (4.11) are a subset of constraints (4.10).

The following proposition is easily seen. Its proof is based on [26].

Proposition 21 *Problem SNDP (resp., VC-SNDP) on a complete graph with positive, non-zero integer edge costs $c \in \mathbb{N}^E_{\geq 1}$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$, and problem SNDP (resp., VC-SNDP) with edge costs all 1 on a general graph (equivalently, the unweighted problem) and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$, are equivalent.*

Proof: Let $G = (V, E)$ be a complete graph with positive, non-zero integer edge costs $c \in \mathbb{N}^E_{\geq 1}$, and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. Replace each edge $e \in E$ with edge cost $c_e > 1$, by a path P_e of length c_e, consisting of $c_e - 1$ Steiner vertices (vertices $i \in V$ with vertex connectivity requirement $r_i = 0$) and c_e edges with edge cost 1. The resulting graph $I = (\tilde{V}, \tilde{E})$ can be made complete by adding in the "missing" edges ij and assigning them appropriately large edge costs. The resulting graph, $\hat{I} = (\tilde{V}, \hat{E})$, is complete and has positive edge costs. Replacing all the paths P_e in the optimal SNDP (resp., VC-SNDP) solution on $\hat{I}$ by the corresponding original edges e yields a feasible SNDP (resp., VC-SNDP) solution on G with equal total edge cost, which is thus an optimal SNDP (resp., VC-SNDP) solution on G. Thus, SNDP (resp., VC-SNDP) with positive and non-zero integer edge costs $c \in \mathbb{N}^E_{\geq 1}$, and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$ can be solved by SNDP (resp., VC-SNDP) with edge costs all 1, and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$. Clearly, the reverse holds. Thus, the two problems are equivalent. ■

Notice that the problem SNDP (resp., VC-SNDP) with edge costs all 1 on a general graph (equivalently, the unweighted problem) and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$, in Proposition 21 is not a special case of metric SNDP (resp., VC-SNDP) with vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$, $i \in V$, since it is on a general graph and not a complete graph. Thus, Proposition 21 and its proof cannot be used to generalize our $\frac{3}{2}$-approximation algorithm result in Corollary 25 to VC-SNDP with positive and non-zero integer edge costs $c \in \mathbb{N}^E_{\geq 1}$ and vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$. A different approach is necessary.

Recall that, in general, $opt(kEC) \leq opt(kVC)$. It can be shown that $opt(metric\ 2EC) = opt(metric\ 2VC)$ [7]. This is an implicit result of Frederickson and Ja'Ja"s shortcutting method ([12]). Notice, however, that this result does not hold in general for higher connectivities, i.e., for $k \geq 3$. For families of counter examples, see ([7], Appendix A) and ([3], Figure 1). However, it can be generalized to include lower connectivities:

Proposition 22 *Let $G' = (V, E)$ be a complete graph with metric edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$. There exists an optimal SNDP solution on G' that is also an optimal VC-SNDP solution on G'. Thus, $opt(metric\ SNDP) = opt(metric\ VC\text{-}SNDP)$.*

Proof: Without loss of generality, assume $|V| \geq 3$ and there are at least two vertices with vertex connectivity 2. Let H be an optimal SNDP solution on G' that is not a feasible VC-SNDP solution. Problems SNDP and VC-SNDP are the same when $r_i \in \{0, 1\}$, i.e., the edge/vertex connectivity holds for all distinct vertex pairs $i, j \in V$ with $r_i \in \{0, 1\}$. Thus, there is at least one $i \in V$ with $r_i = 2$ for which there exists a $j \in V$ with $r_j = 2$ such that there are at least two edge-disjoint paths, P_1, P_2, connecting i and j in H, but less than two internally vertex-disjoint paths connecting i and j in H. Thus, P_1 and P_2 are not internally vertex-disjoint, and there is a "cut" vertex $v \in V$, $v \neq i$ and $v \neq j$, that is in both P_1 and P_2. Let $u, w \in V$ be such that uv and wv are in P_2. Let P_{1a} and P_{1b} be the sections of P_1 between i and v and between v and j, respectively. Let P_{2a} and P_{2b} be the sections of P_2 between i and u and between w and j, respectively. Remove edges uv and vw from H, and add edge uw to form the subgraph H' (note that this edge exists since G' is a complete graph). Any path that uses edges uv and vw in H now uses edge uw in H'. Thus, all "local" edge connectivities r_{pq}, $q, p \in V \backslash \{v\}$ remain unchanged. For example, path P_1 remains the same, but path P_2 becomes $P_{2a}uwP_{2b}$. Thus, there still exist at least two edge-disjoint paths between i and j. Also, there are still at least two edge-disjoint paths between i and v; for instance, paths P_{1a} and $P_{2a}uwP_{2b}, P_{1b}$. Similarly, there are still at least two edge-disjoint paths between j and v. Since $r_v \leq 2$, all "local" edge-connectivities r_{qv}, $q \in V\backslash\{v\}$, are preserved. See Figure 4.13.

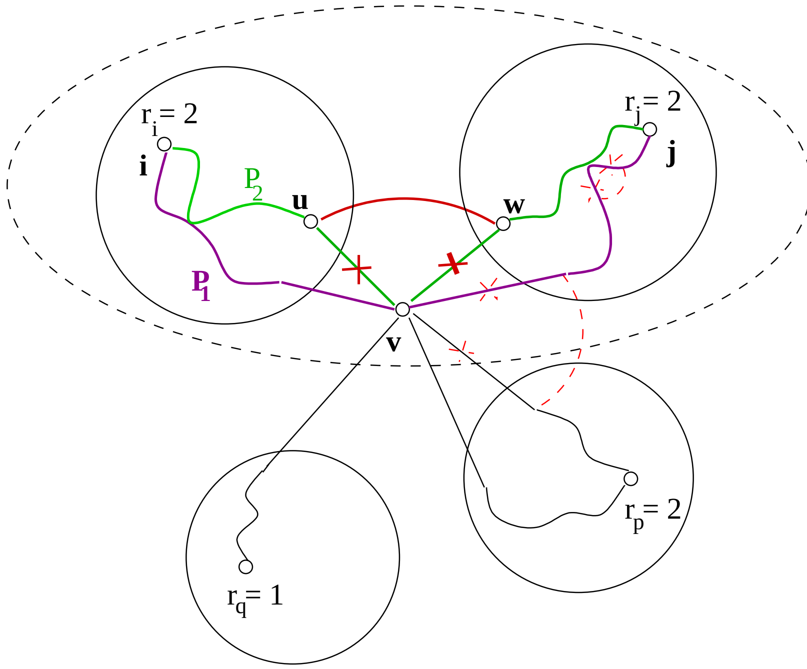

Figure 4.13: Illustration of proof of Proposition 22.

Thus, H' is a feasible SNDP solution. Moreover, since the edge costs are metric, and the cost of H is optimal, H' is an optimal SNDP solution. Repeat for all such internally "shared" vertices of P_1 and P_2. We obtain an optimal SNDP solution in which i and j have two internally vertex-disjoint paths connecting them. Repeating for all $i, j \in V$, $r_i = r_j = 2$, such that there is only one internally vertex-disjoint path connecting i and j in H', we obtain an optimal SNDP solution that is also a feasible VC-SNDP solution, $r_i \in \{0, 1, 2\}$, $i \in V$. Since $opt(SNDP) \leq opt(VC\text{-}SNDP)$, this solution is an optimal VC-SNDP solution for the case with $r_i \in \{0, 1, 2\}$, $i \in V$. ■

Theorem 23 *Given a complete graph $G = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$. All optimal solutions for multi-kVC are either optimal solutions for kVC or have multi-edges only on edges with edge cost 0. In particular, $opt(\text{multi-}kVC) = opt(kVC)$.*

Proof: Let H be an optimal solution to multi-kVC on G. If H is simple, we are done. Let $e \in E(H)$ be an edge that has more than one copy in H. Let $u, v \in V$. There are at least k internally vertex-disjoint paths from u to v in H. If more than one of them contains e, then they are no longer internally vertex-disjoint. Thus, at most one of them contains e. This is true for all distinct vertex pairs in V. Thus, removing all the multiple copies of edge e from H (and leaving just a single copy of e in H) yields a (multi)-subgraph H' of G that is k-vertex connected and whose total cost is less than or equal to the total cost of H. Since H is an optimal solution to multi-kVC on G, the total cost of H' equals the total cost of H. Thus, the edge cost of e must be 0. Repeating this argument for each multi-edge of H, we obtain a simple subgraph $\hat{H}$ of G with total cost equal to the total cost of H.

Hence, every optimal solution for multi-kVC is either an optimal solution for kVC (i.e., it is simple), or its multi-edges only involve edges with edge cost 0; and $opt(\text{multi-}kVC) = opt(kVC)$. ■

Corollary 24 *Given a complete graph $G = (V, E)$ with positive edge costs $c \in \mathbb{R}^E_{>0}$. Problems multi-kVC and kVC are equivalent.*

Proof: Follows directly from Theorem 23. ■

4.5.1 Approximation Algorithm for Metric VC-SNDP with $r_i \in \{0, 1, 2\}$

Applying the construction in the proof of Proposition 22 to any feasible solution to metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, we obtain in polynomial time, a feasible solution to metric VC-SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, with equal total edge cost. Thus, any α-approximation algorithm for metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, can be turned into an α-approximation algorithm for metric VC-SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$. In the case where we have a feasible metric 2EC solution, as we do from step (A5) in Section 4.4.1, this construction reduces to Frederickson and Ja'Ja"s shortcutting method [12], becoming as follows:

A6. For each cut vertex v of H: Take edges uv and vw in $\tilde{E}$ such that u and w are in different blocks (maximal two-vertex connected subgraphs of the graph); replace uv and vw by $uw \in E$. Repeat until all the blocks of the subgraph are merged and no cut vertices remain.

Step (A6) transforms, at no increase in total edge cost, a feasible metric 2EC solution into a 2-vertex connected spanning simple subgraph. We can generalize this to apply to metric VC-SNDP with vertex connectivity requirements $r_i \in \{0, 1, 2\}$:

Corollary 25 *Given a complete graph $G' = (V, E)$, $|V| \geq 3$, with metric edge costs $c' \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$, the approximation algorithm from ([4], Algorithm A) (with $r_m = r_l = 2$), with the addition of steps (A5) and (A6), is a $\frac{3}{2}$-approximation algorithm for metric 2VC (for this case, steps (A0) and (A4) may be omitted), and for metric VC-SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, respectively.*

Proof: Let H be the heuristic metric 2EC solution returned by the approximation algorithm in ([4], Algorithm A) (with $r_m = r_l = 2$) with the addition of step (A5). If

H has no cut vertices, then H is 2-vertex connected and we are done. Let $v \in V$ be a cut vertex of H. Let X and Y be two blocks of H which share v. Thus, v lies on a cycle C_X in X and a cycle C_Y in Y. Let $u \in V$ and $w \in V$ be adjacent to v on C_X and C_Y, respectively. Remove the edges uv and vw from H and add the edge uw to H to form the subgraph $\hat{H}$ on G'. The blocks X and Y are now one block. Since the degree of v in H was at least *2(the number of connected components of* $H - \{v\}$*)* ≥ 4, the degree of v in $\hat{H}$ is at least *2(the number of connected components of* $H - \{v\}$*)*$-2 \geq 2$. All other degrees remain unchanged. Thus, $\hat{H}$ is 2-edge connected and is a spanning subgraph of G'. Repeat this until all the blocks sharing v are merged into one block, i.e., until v is no longer a cut vertex in $\hat{H}$. Since the edge costs are metric, the cost of $\hat{H}$ is less than or equal to the cost of H. Repeat the above for all the cut vertices of H. The resulting subgraph, N, of G' is spanning, 2-vertex connected, and has total edge cost no more than that of H. Since step (A6) can be done in polynomial time, and Algorithm A has a $\frac{3}{2}$ approximation guarantee for this case, this yields a $\frac{3}{2}$-approximation algorithm for metric 2VC.

From the proof of Theorem 20, the heuristic solution of metric SNDP, $r_i \in \{0, 1, 2\}$, $i \in V$, that is obtained from the approximation algorithm in ([4], Algorithm A) (with $r_m = r_l = 2$) with the addition of step (A5), is a spanning subgraph that is 2-edge connected and simple. Thus, step (A6) can be applied to the heuristic solution of metric SNDP, $r_i \in \{0, 1, 2\}$, $i \in V$, that is obtained by that approximation algorithm (i.e., the algorithm consisting of steps (A0) to (A5)). From the argument above, the resulting subgraph of G' is spanning, 2-vertex connected, and has total edge cost no more than that of H. Thus, this is a heuristic solution for metric VC-SNDP, $r_i \in \{0, 1, 2\}$, $i \in V$. Since this can be done in polynomial time, we have a $\frac{3}{2}$-approximation algorithm for metric VC-SNDP, $r_i \in \{0, 1, 2\}$, $i \in V$. ■

The proof of our $\frac{3}{2}$-approximation algorithm for metric 2VC provides an alternate, simpler, and more compact proof for Frederickson and Ja'Ja"s result.

The best constant factor approximation algorithm for VC-SNDP with general edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$, is currently a 2-approximation algorithm [11]. For unweighted (equivalently, edge costs all 1) VC-SNDP on a general graph with vertex connectivity requirements $r_i \in \{1, 2\}$, $i \in V$, there is a $\frac{3}{2}$-approximation algorithm [26]. Thus, our $\frac{3}{2}$-approximation algorithm for metric VC-SNDP, $r_i \in \{0, 1, 2\}$, $i \in V$, extends the result of [26].

Integrality Gap, Metric VC-SNDP with $r_i \in \{0, 1, 2\}$

Our integrality gap results can be extended to an upper bound on integrality gap of the linear programming relaxation of metric VC-SNDP, $r_i \in \{0, 1, 2\}$, $i \in V$.

Lemma 26 *Let $G' = (V, E)$, $|V| \geq 3$, be a complete graph with metric edge costs $c' \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$, and let G have at least 3 vertices with $r_i = 2$. Then the integrality gap for the linear programming relaxation of VC-SNDP on G', $r_i \in \{0, 1, 2\}$, $i \in V$, is less than or equal to the integrality gap for the linear programming relaxation of multi-SNDP on G', $r_i \in \{0, 1, 2\}$, $i \in V$.*

Proof: By Lemma 3, and since the linear programming formulation for SNDP is a relaxation of the linear programming formulation for VC-SNDP,

$$opt(\textit{multi-SNDP}_{LP} \textit{ on } G') \leq opt(\textit{SNDP}_{LP} \textit{ on } G') \leq opt(\textit{VC-SNDP}_{LP} \textit{ on } G').$$

By Theorem 14, $opt(\textit{multi-SNDP on } G') = opt(\textit{SNDP on } G')$, for $r_i \in \{0, 1, 2\}$, $i \in V$, and by Proposition 22, $opt(\textit{SNDP on } G') = opt(\textit{VC-SNDP on } G')$, for $r_i \in \{0, 1, 2\}$, $i \in V$. We therefore have that

$$\frac{opt(\textit{VC-SNDP on } G')}{opt(\textit{VC-SNDP}_{LP} \textit{ on } G')} \leq \frac{opt(\textit{multi-SNDP on } G')}{opt(\textit{multi-SNDP}_{LP} \textit{ on } G')},$$

for $r_i \in \{0, 1, 2\}$, $i \in V$. ■

The following lemma is obtained by combining Lemma 26 and Theorem 6.

Lemma 27 *Given a complete graph $G' = (V, E)$, $|V| \geq 3$, with metric edge costs $c' \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$, the integrality gap for the LP relaxation of metric VC-SNDP, $r_i \in \{0, 1, 2\}$, $i \in V$, is at most $\frac{3}{2}$.*

4.5.2 2VC and 2EC on a 2-vertex Connected Multi-graph

Let G^v be a 2-vertex connected multi-graph with non-negative edge costs. In this section, we examine the following question: When we are finding an optimal solution for 2VC or 2EC on G^v, when can we consider, without loss of generality, just the underlying simple graph of G?

Proposition 28 *Let G^v be a 2-vertex connected multi-graph with non-negative edge costs and $|V| \geq 3$. Every optimal solution to 2VC on G^v is simple.*

Proof: Let K be an optimal solution of 2VC on G^v. Suppose K has a pair of parallel edges e_1, e_2 between vertices $a, b \in V$. Since K is 2-vertex connected, there exist at least two internally vertex-disjoint paths, P_1 and P_2, in K from a to b. Assume e_1 is in P_1 and e_2 is in P_2. If $\{P_1, P_2\}$ is a maximal set of pairwise internally vertex-disjoint paths from a to b, then $\{e_1, e_2\}$ form a cut in K, and thus at least one of a and b is a cut vertex. This is a contradiction since K is 2-vertex connected. Thus, there exists another path P from a to b in K such that $\{P_1, P_2, P\}$ is a set of pairwise internally vertex-disjoint paths. Note that P can not include e_1 or e_2. Thus, either e_1 or e_2 can be removed from K without decreasing the vertex connectivity of K. In doing this, we have decreased the total edge cost of K. This is a contradiction. Therefore, K can not be a multi-subgraph, i.e., K must be a simple subgraph of G^v. ■

Proposition 28 shows that, for 2VC on G^v, we can delete the parallel edges of highest cost (since they will never be used), and solve 2VC on the underlying simple subgraph of G^v.

By [30], in a 2-vertex connected multi-graph with uniform edge costs, there is an optimal solution for 2EC that is simple. However, when considering 2EC on G^v

(where the edge costs are not uniform), we cannot consider just the underlying simple subgraph – the cost of an optimal solution of 2EC on G^v can be strictly less than that on the underlying simple subgraph of G^v. For example, see the multi-edge "spaceship" example in Figure 4.14; showing a 2-vertex connected multi-graph on which there is a cheaper optimal solution to 2EC than on the underlying simple subgraph.

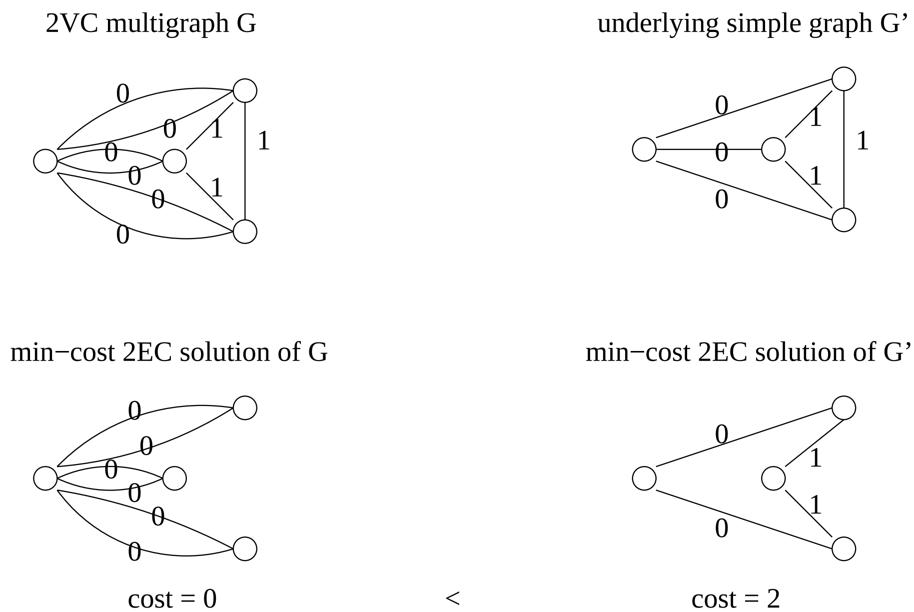

Figure 4.14: An example where an optimal solution of 2EC on a 2-vertex connected multi-graph is cheaper than an optimal solution of 2EC on the corresponding underlying simple subgraph.

To better understand why this occurs, suppose the min-cost solution H of G contains two multiple edges ab, call them e_1 and e_2. If e_1 and e_2 are the only edge-disjoint paths between a and b in G, then neither one can be removed for the solution to still remain feasible (i.e., 2EC). Therefore we can not just consider the underlying simple graph of G. On the other hand, if $\{e_1, e_2\}$ is not a maximal set of edge-disjoint paths from a to b, then there exists a path P from a to b including neither e_1 nor e_2. If $cost(P) < cost(ab)$ (non-metric cost), then the solution would have used P

instead of e_1 or e_2. If $cost(P) = cost(ab)$, then we might be able to delete e_1 or e_2 and replace it by P (if x_e can be > 1; otherwise, if not, then H would not have parallel edges). If $cost(P) > cost(ab)$, then we can't remove either e_1 or e_2 and replace it by P to get a solution of same value. Therefore, we can't in general change the optimal min-cost 2EC multi-subgraph into a simple subgraph of the same value. Thus, for 2EC, when the edge costs are not uniform we can not consider just the underlying simple subgraph of a 2-vertex connected multi-graph G^v.

Thus, when finding an optimal solution for 2VC (general non-negative or uniform edge costs) or for 2EC (uniform costs only) on a 2-vertex connected multi-graph, we can, without loss of generality, consider just the underlying simple graph of G. This is because, as we have shown, all the respective solutions will be simple. This is not, however, possible to do for optimal solutions of the 2EC with general non-negative edge costs, since the optimal solution of 2EC on the 2-vertex connected multi-graph can have total edge cost less than that of an optimal solution of 2EC on the corresponding underlying simple graph.

4.6 Conclusion

In this paper, we presented extensions, and applications, of the results of our previous work in [4] to related problems, and investigated cases where the optimal values of multi-SNDP and SNDP are equal. We found an upper bound of $\frac{3}{2}$ on the integrality gaps of the LP relaxations of metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, and of metric VC-SNDP $r_i \in \{0, 1, 2\}$, $i \in V$. This improves on the previously known upper bound of $\frac{3}{2}$ for the integrality gap of the LP relaxation of metric 2EC [1]. Our $\frac{3}{2}$ approximation guarantee and integrality gap bound applies to the Steiner tree problem with either uniform or metric edge costs, and improves the previous best known integrality gap bound for these cases [5].

By making certain modifications to our multi-SNDP approximation algorithm

[4], we presented a $\frac{3}{2}$-approximation algorithm for metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, and metric VC-SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$. This is an improvement on the previously best known integrality gap bound and approximation guarantee for this special case of metric VC-SNDP, which was 2 [11]. This also extends the known results of a $\frac{3}{2}$-approximation algorithm for metric 2EC [12], a $\frac{3}{2}$-approximation algorithm for VC-SNDP with $r_i \in \{1, 2\}$, $i \in V$, on a general graph with edge costs all 1 [26]; and a $\frac{3}{2}$-approximation algorithm for metric 2VC [12].

Lastly, it is interesting to take note of a problem that is related to kVC, and which is somewhat of an intermediate problem between kVC and VC-SNDP. A graph is called *k-out-connected from r* if there exist k pairwise internally vertex-disjoint paths from a given vertex r to every other vertex in the graph [2]. Given a complete graph, $G = (V, E)$, with non-negative edge costs $c \in \mathbb{R}^E$, and a vertex $r \in V$, the *k vertex out-connected-from-r spanning subgraph problem* (kVC$\{r\}$) is the problem of finding a minimum cost spanning subgraph of G that has k pairwise internally vertex-disjoint paths between r and v, for all $v \in V$, $v \neq r$. Clearly, a feasible solution to kVC also has k edge-disjoint paths from r every other vertex in the graph. Thus, kVC is a special case of kVC$\{r\}$. On the other hand, a feasible solution to kVC$\{r\}$ is not necessarily k-vertex connected. For example, a k-out-connected-from-r spanning subgraph may contain cut sets, and there may be cut sets even when the degree of r in such a subgraph is exactly k. For example, consider the complete graph on 8 vertices, with edge cost 1 on the edges that are shown in Figure 4.15, and edge cost 20 on all other edges. There is a feasible solution to 4VC$\{r\}$ on this graph that, although not containing any cut vertices, contains a cut set of size 3 (and in which the degree of r is 4). See Figure 4.15. Thus, a feasible solution to kVC$\{r\}$, although vertex connected, is not necessarily k-vertex connected (even when the degree of r in the solution is exactly k). An exception is when $k = 2$, in which case 2VC$\{r\}$ is precisely 2VC.

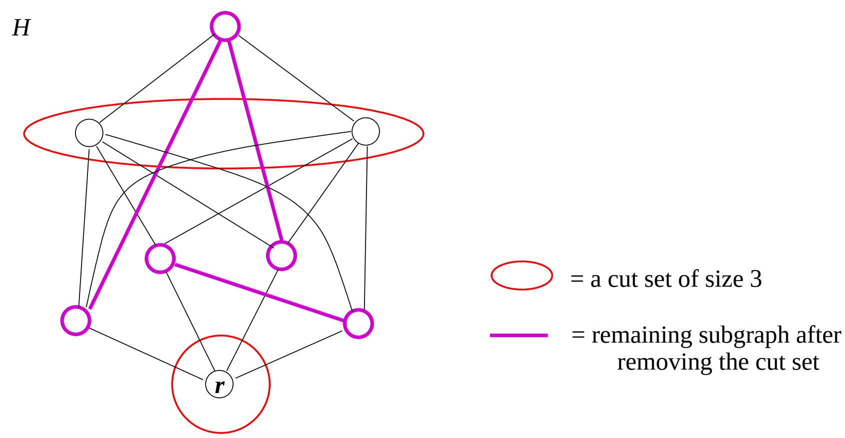

Figure 4.15: A feasible solution, H, to 4VC$\{r\}$ on a complete graph with edge costs 1 on the edges shown, and edge cost 20 on all other edges. Notice that H has no cut vertices, the degree of r in H is 4, and H has a cut set of size 3 (therefore H is not 4-vertex connected).

The following is an ILP formulation of kVC$\{r\}$ on G:

$$
\begin{aligned}
&\text{minimize} \quad cx && (4.14)\\
\text{subject to} \quad x(\delta(S)) &\geq k, \quad \text{for all } \emptyset \subset S \subset V, && (4.15)\\
x(\delta_{G[V\setminus U]}(S)) &\geq 1, \quad \text{for all } \emptyset \subset U \subset V \text{ such that } |U| = k-1, &&\\
&\qquad \text{and for all } \emptyset \subset S \subset V\setminus U \text{ such that } r \in S, && (4.16)\\
x_e &\geq 0, \quad \text{for all } e \in E, && (4.17)\\
x_e &\quad \text{integer.}
\end{aligned}
$$

Notice that this ILP is very similar to the ILP for kVC. The difference is that the constraints (4.16) are a subset of the constraints (4.7), since we only need k pairwise internally vertex-disjoint paths between r and every other vertex, not between all distinct pairs of vertices. Indeed, the optimal value of kVC$\{r\}$ provides a lower bound on the optimal value of kVC, since kVC$\{r\}$ is a relaxation of kVC (relaxing the node cut constraints (4.7)). In other words, if $G = (V, E)$ is a complete graph with non-negative edge costs $c \in \mathbb{R}^E$ and $k \in \mathbb{N}$, $k \geq 2$, then $opt(kVC\{r\}) \leq opt(kVC)$.

Problem kVC$\{r\}$ is an interesting problem to consider, since it is a step further than kVC, but it is not as involved as VC-SNDP. This problem is also of interest since the 2-approximation algorithm for 3VC in [2] calls a subroutine that returns a feasible solution of kVC$\{r\}$ with total cost at most twice the optimal value [9]. If the approximation guarantee of 2 for kVC$\{r\}$ can be improved, then the approximation guarantee of 2 for 3VC can also be improved. Thus, future work needs to be done on kVC$\{r\}$ to investigate if the current approximation guarantee of 2 for kVC$\{r\}$ can be improved.

Bibliography

[1] A. Alexander, S. Boyd, P. Elliott-Magwood, On the Integrality Gap of the 2-Edge Connected Subgraph Problem, Technical Report TR-2006-04, S.I.T.E., University of Ottawa, Ottawa, 2006.

[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] D. Bienstock, E. Brickell, and C. Monma, On the Structure of Minimum-weight k-connected Spanning Networks, SIAM J. Discrete Math., 3 No.3 (1990) 320329.

[4] 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).

[5] J. Byrka, F. Grandoni, T. Rothvoβ, L. Sanità, An Improved LP-based Approximation for Steiner Tree, STOC 2010, Cambridge, Massachusetts, USA (2010).

[6] R. Carr, R. Ravi, A New Bound for the 2-Edge Connected Subgraph Problem, Lecture Notes in Computer Science 1412 (1998) 112-125.

[7] J. Cheriyan, A. Vetta, Approximation Algorithms for Network Design with Metric Costs, SIAM J. Discrete Math. 21 No.3 (2007) 612-636.

[8] N. Christofides, Worst Case Analysis of a New Heuristic for the Traveling Salesman Problem, Report 388, Graduate School of Industrial Administration, Carneigie Mellon University, Pittsburgh, 1976.

[9] Y. Dinitz, Z. Nutov, A 3-Approximation Algorithm for Finding Optimum 4,5-Vertex-Connected Spanning Subgraphs, Journal of Algorithms 32 (1999) 31-40.

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

[11] L. Fleischer, K. Jain, D. Williamson, Iterative Rounding 2-approximation Algorithms for Minimum-cost Vertex Connectivity Problems, Journal of Computer and Systems Sciences 72 (2006) 838-867.

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

[13] H.N. Gabow, M.X. Goemans, D.P. Williamson, An Efficient Approximation Algorithm for the Survivable Network Design Problem, Mathematical Programming 82 (1998) 13-40.

[14] M.R. Garey, R.L. Graham, D.S. Johnson, The Complexity of Computing Steiner Minimal Trees, SIAM J. Appl. Math. 32 (1977) 835-859.

[15] 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).

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

[17] M. Grötschel, C.L. Monma, Integer Polyhedra Arising from Certain Network Design Problems with Connectivity Constraints, SIAM Journal Discrete Mathematics 3 No.4 (1990) 502-523.

[18] M. Grötschel, C.L. Monma, M. Stoer, Design of Survivable Networks, in: M.O. Bell, T.L. Magnant, C.L. Monma, G.L. Nemhauser, eds, Network Models (Handbooks in OR & MS, Vol.7), North-Holland, 1995.

[19] M. Grötschel, C.L. Monma, M. Stoer, Polyhedral Approaches in Network Survivability, DIMACS Series in Discrete Mathematics and Theoretical Computer Science 5 (1991) 121-141.

[20] M. Goemans, Y. Myung, A Catalog of Steiner Tree Formulations, Networks 23 (1993) 19-28.

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

[22] S. Khuller, Approximation Algorithms for Finding Highly Connected Subgraphs, in: Approximation Algorithms for NP-hard Problems, PWS Publishing Company, Boston, (1997) 236265.

[23] J. Könemann, D. Pritchard, K. Tan, A Partition-based Relaxation for Steiner Trees, Math. Programming Series A (2009).

[24] G. Kortsarz, Z. Nutov, Approximating Minimum-Cost Connectivity Problems, in: T.F. Gonzalez, ed., Handbook of Approximation Algorithms and Metaheuristics, Chapman & Hall, 2007.

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

[26] P. Krysta, Approximating Minimum Size $\{1, 2\}$-connected Networks, Discrete Applied Mathematics 125 (2003) 267-288. First appeared in: P. Krysta, A. Kumar, Approximation Algorithms for Minimum Size 2-Connectivity Problems, STACS (2001) 431-442.

[27] M. Mihail, D.S. Shallcross, N. Dean, M. Mostrel, A Commercial Application of Survivable Network Design: ITP/INPLANS CCS Network Topology Analyzer, Chapter 32, Proceedings of the seventh annual ACM-SIAM symposium on Discrete algorithms (1996) 279-287.

[28] C.L. Monma, B.S. Munson, W.R. Pulleyblank, Minimum-weight Two-connected Spanning Networks, Mathematical Programming 46 (1990) 153-171.

[29] G. Robins, A. Zelikovsky, Tighter Bounds for Graph Steiner Tree Approximation, SIAM J. Discrete Math 19 No.1 (2005) 122-134.

[30] S. Vempala, A. Vetta, Factor $\frac{4}{3}$ Approximations for Minimum 2-Connected Subgraphs, Proceedings of the 3rd International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 2000, Springer, Berlin, (2000) 262-273.

Chapter 5

Linear Programming Relaxations of Steiner Trees

• Amy Cameron. LP Relaxations of Steiner Trees, Technical Report, University of Ottawa, 2012.

Work of Amy Cameron under the supervision of Sylvia Boyd. All results are work of Amy Cameron, with minor editorial comments from Sylvia Boyd.

Abstract

The quality of the results that one can obtain for a given network design problem often depends on its integer linear programming formulation, in particular, on its linear programming relaxation. This paper investigates different formulations for the Steiner tree problem. We examine known integer linear programming formulations for the Steiner tree problem, and the bounds on the integrality gap of their respective linear programming relaxations. We propose two new formulation for the Steiner tree problem and investigate their strength. The first proposed formulation is shown to not be very promising. The second proposed formulation has a lower bound of 8/5 for the integrality gap of its linear programming relaxation.

5.1 Introduction

Given a network consisting of critical centres, intermediate centres, and possible links connecting the centres, we seek to construct a network that connects the critical centres. When each link has a cost associated to building it, we seek to construct such a network in the cheapest way possible. This is known as the Steiner tree problem. Since any, all, or none of the intermediate centres may be used in constructing the network, we can often construct a connected network with smaller cost than if the network could not use any of the intermediate centres.

The Steiner tree problem originally arose in geometry, where one wished to find a set of lines of minimum total length that connected a given set of points in a Euclidean plane [14]. These lines could intersect each other at points that were not part of the given set of points. Such intersection was useful since it could frequently reduce the total length of the connecting lines. This problem was eventually generalized to networks. For a survey of the early work on the Steiner problem, see [21].

From its beginning in geometry, the Steiner tree problem has developed into an important problem in network design, with numerous important applications. For example, designing a minimum cost road network that connects rural towns, given a finite number of permissible intersections (based on terrain, development, and other considerations) can be modelled as a Steiner network problem [3]. The Steiner tree problem also has applications to designing communication, snow removal, and drainage systems [3], and is applicable to organizational design [29] and phylogenetic tree reconstruction in biology [22]. The Steiner tree problem is highly applicable to VLSI (Very Large Scale Integration, or Microelectronics) layout [31]; such as optimally designing wire layouts in circuit boards, microprocessors, etc.

We now formally define the Steiner tree problem: Given an (undirected) graph $G = (V, E)$ and a subset of vertices $T \subseteq V$, a *Steiner tree* is a connected subgraph of G that includes the vertices in T (and may also include some, all, or no vertices from $V \backslash T$) and that does not contain any cycles. Let $|V| = n$, $n \in \mathbb{N}_{>0}$. Vertices in T are called *terminal* (or *required*) *vertices*, and non-terminal vertices in $V \backslash T$ are called *Steiner vertices*. Given non-negative edge costs $c \in \mathbb{R}^E$, the *Steiner tree problem* (denoted ST), sometimes also called the *Steiner minimal tree*, is the problem of finding a minimum cost Steiner tree of G; i.e., a minimum cost tree that contains all the terminal vertices T and any subset (including the empty set) of the Steiner vertices. Notice that there is never a need to use more than one copy of any edge in a solution to ST, since any multiple copy of an edge can always be removed without disconnecting that solution. When the edge costs $c \in \mathbb{R}^E$ of a graph $G = (V, E)$ are non-negative and satisfy the triangle inequality $c_{uw} \leq c_{uv} + c_{vw}$ for all distinct vertices $u, v, w \in V$, the costs are called *metric*. If the edge costs are metric (or unweighted), then the problem is referred to as the metric Steiner tree problem.

The following definitions will be used throughout this paper: Given a graph $G = (V, E)$, let $\delta(S)$ denote the set of edges $\{uv \in E : u \in S, v \notin S\}$, i.e, the set of edges with exactly one end in S. Let $\gamma(S)$ be the set of edges $\{uv \in E : u, v \in S\}$, i.e., the set of edges with both ends in S. The set $\delta(S)$ is called a *cut*. A subset $S \subset V$ *separates* vertices of T if $S \cap T \neq \emptyset$ and $S \cap T \neq T$, i.e., there is at least one terminal vertex in S and at least one terminal vertex not in S. A *partition* of V is a set $P = (P_1, P_2, \ldots, P_p)$ such that $P_i \subseteq V$ for $i = 1, 2, \ldots, p$, $V = P_1 \cup P_2 \cup \ldots \cup P_p$, and $P_i \cap P_j = \emptyset$ for $i, j \in \{1, 2, \ldots, p\}$, $i \neq j$. 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. A *Steiner partition* is a partition $\Pi = (V_1, V_2, \ldots, V_{k_\Pi})$ of V, where each part V_i contains at least one terminal vertex, i.e., $V_i \cap T \neq \emptyset$ for $i = 1, 2, \ldots, k_\Pi$. A Steiner partition with $k_\Pi = 2$ is called a *Steiner cut*. Let G/Π be the multi-graph obtained by identifying (or "shrinking") each part V_i in Π as a vertex v_i', $i = 1, 2, \ldots, k_\Pi$, and keeping the edges of G whose ends lie in different parts of Π. This "shrunk" graph G/Π has k_Π vertices, and edge set E_Π corresponding to the set of edges of G whose ends lie in different parts of Π. Notice that for this paper, without loss of generality, we can convert G/Π into a simple graph by removing all but the lowest cost edge in every set of parallel edges. Let $G[V_i]$ be the subgraph of G induced by the vertices V_i; in other words, $G[V_i]$ is the induced subgraph consisting of the vertices V_i and the edges $\gamma(V_i)$. For basic definitions in polyhedral theory, see [13].

There are two well known special cases of ST that are solvable in polynomial time. The *minimum spanning tree problem* (MST) is the special case of ST where all the vertices are terminal vertices, i.e., where $T = V$. The *shortest s-t path problem* is the special case of ST where the set of terminal vertices T is simply the subset $\{s, t\}$. Although these two special cases of ST can be solved in polynomial time[1], the Steiner tree problem is NP-hard [16, 21, 24].

Since 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. Interestingly, not only is ST NP-hard, but it has also been shown that it is NP-hard to even obtain an approximation guarantee better than $\frac{96}{95}$ for ST [9].

An important and related concept to approximation guarantees is that of integrality gaps. Given a minimization problem P, and its integer linear program (ILP) Q, a linear program is a *linear programming (LP) relaxation*, Q_{LP}, of Q if every feasible solution of Q is also a feasible solution of Q_{LP}. In this paper, we refer to the LP relaxation of an ILP as the linear programming relaxation formed by relaxing the integer constraints. Let $opt(Q)$ be the value of an optimal solution to Q, and let $opt(Q_{LP})$ be the value of an optimal solution to Q_{LP}. The optimal value of Q_{LP}

[1] For example, MST can be solved by Prim's or Kruskal's algorithm; and the shortest path problem can be solved by Dijkstra's algorithm [8, 13].

is at most the optimal value of Q, and thus provides a lower bound on the optimal value of Q. The question naturally arises as to how big a difference, or "gap," can possibly exist between $opt(Q)$ and $opt(Q_{LP})$. This is where the notion of integrality gap enters.

Formally, the *integrality gap* of a linear programming relaxation of a minimization problem P is the largest ratio $opt(Q)/opt(Q_{LP})$ over all possible non-negative, and all not identically 0, objective cost functions. If it can be proved that this largest ratio has value β (where β is some constant value), the linear programming relaxation is said to have a β *integrality gap*; indicating that, in the worst possible case, $opt(Q)$ is no more than $\beta\ opt(Q_{LP})$. Thus, the integrality gap provides a measure of the quality of the lower bound provided by Q_{LP} for Q. If the integrality gap of a problem's LP relaxation is 1, then that LP relaxation is said to be *exact*, and there is always an optimal solution of that LP formulation that is integer.

It is important to note that the integrality gap of a LP relaxation depends on the formulation of the linear program – one formulation for a given problem may give a better integrality gap than a different formulation for the same problem. This can be seen simply from polyhedral theory: Since the underlying polyhedron of Q_{LP} contains the underlying polyhedron of Q, taking a facet of the polyhedron associated with Q and adding it as a constraint to Q_{LP} reduces the feasible area of the LP relaxation, thus giving a stronger LP relaxation and potentially reducing the difference, or "gap", between the value of an optimal solution to Q and the value of an optimal solution to the new LP relaxation. Given a specific graph $\hat{G}$ with non-negative edge costs, we refer to the *integrality ratio* as the ratio of the cost of an optimal solution to Q on $\hat{G}$ over the cost of an optimal solution to Q_{LP} on $\hat{G}$. Every integrality ratio gives a lower bound on the integrality gap of Q_{LP}.

Approximation guarantees and integrality gaps, although distinct concepts, are closely related: A polynomial-time constructive proof for a bound α on the integrality gap of an LP relaxation Q_{LP} provides an α-approximation algorithm for problem P.

In other words, if we can construct, in polynomial-time, a feasible solution H (with total edge cost $cost(H)$) to Q such that $cost(H) \leq \beta\ opt(Q_{LP})$, then we have both

$$opt(Q) \leq cost(H) \leq \beta\ opt(Q_{LP}); \quad \text{i.e., } \frac{opt(Q)}{opt(Q_{LP})} \leq \beta,$$

and

$$cost(H) \leq \beta\ opt(Q_{LP}) \leq \beta\ opt(Q); \quad \text{i.e., } \frac{cost(H)}{opt(Q)} \leq \beta,$$

and hence we have an upper bound of β for the integrality gap for the LP relaxation Q_{LP}, as well as a β-approximation algorithm for the problem P. In addition to this connection, many approximation algorithms are based on an LP relaxation, with the algorithm returning a feasible ILP solution whose cost is close to the cost of an optimal LP solution [27] (for example, the primal-dual method). Thus, obtaining an LP relaxation that has a small (close to 1) integrality gap, can aid in obtaining better approximation algorithms for P. In fact, it has been observed that often the best known approximation guarantee and the best known lower bound for the integrality gap for a problem match [6].

The Steiner problem is part of an important larger class of network design problems, to which we will refer in this paper. This general network problem, stated by Jain [23], is as follows: Given a multi-graph[2] $G' = (V, E)$ with non-negative edge costs $c \in \mathbb{R}^E$, and a weakly supermodular cut requirement function $f : 2^V \longrightarrow \mathbb{Z}$, find a minimum cost spanning multi-subgraph of G' such that there are at least $f(S)$ edges in every cut $\delta_{G'}(S)$, for all $S \subseteq V$. Notice that ST is a special case of Jain's general network problem – simply set $f(S) = 1$ if $S \subset V$ separates vertices of T, and $f(S) = 0$ otherwise. See Figure 5.1. For the special case of MST, Jain's LP relaxation for his general network problem gives an integrality gap that is asymptotically close to 2 [23, 36]. Thus, Jain's LP relaxation for his general network problem has an

[2] A graph G is a *multi-graph* if it can have multiple copies of an edge between at least one pair of distinct vertices in G. In a weighted multi-graph, multiple copies of edges are considered to be distinct edges.

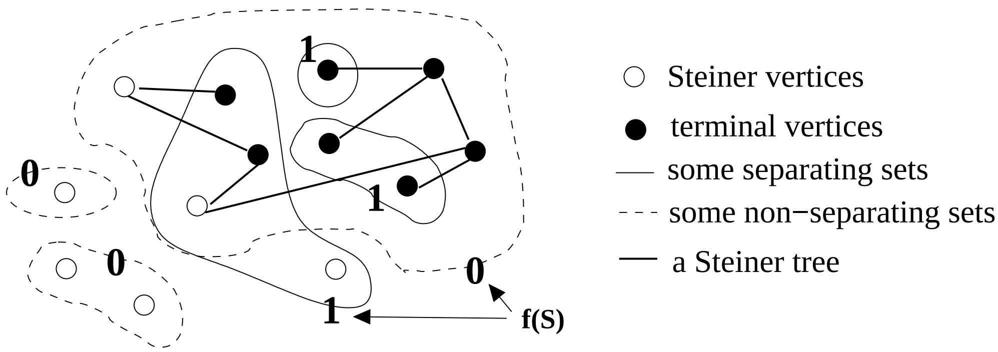

Figure 5.1: An example of a Steiner tree, along with some separating sets and their corresponding cut requirements $f(S)$.

integrality gap asymptotically close to 2 for Steiner trees, as well as for his general network problem.

As mentioned earlier, the integrality gap of the LP relaxation of a problem depends on the formulation of the linear program – one linear programming formulation for a given problem may give a smaller integrality gap than another linear programming formulation for the same problem. For example, for the special case of MST, Jain's LP relaxation of his general network problem [23] gives an integrality gap that is asymptotically close to 2, however, MST can be solved exactly in polynomial time. In particular, there are well known LP relaxations of MST that have an integrality gap of 1. Clearly, Jain's LP relaxation is not the best LP relaxation to use when examining MST.

In this paper, we examine LP relaxations for ST, and their corresponding integrality gaps. Since ST is NP-hard, we know that, unless $\mathcal{P}$=$\mathcal{NP}$, we cannot get an exact LP relaxation (i.e., with integrality gap 1) for ST like we can for its special case MST. Our goal, then, is to obtain the best possible LP relaxation for ST; in other words, to find an LP relaxation for ST that has an integrality gap as close to 1 as possible. Thus, we investigate various LP formulations for ST to see if we can find one which yields a smaller integrality gap than is currently known.

There are more results on approximation algorithms for ST and their approximation guarantees than there are on integrality gap of LP relaxations for ST. A very basic 2-approximation algorithm for ST simply finds the minimum spanning tree on the complete graph consisting of the terminal vertices T, with edge costs the cost of a minimum cost path between i and j in G, for all $i, j \in T$ [17, 28]. This was improved to a $2(1 - \frac{1}{|T|})$-approximation algorithm which, starting with the tree T_1 consisting of a vertex $v_1 \in T$, constructs the tree T_{i+1} by adding in a vertex in $T \backslash V(T_i)$ that has the minimum cost path between it and one of the vertices in T_i, for all $i = 1, 2, \ldots, |T|$ (this is a generalization of Prim's algorithm for MST) [35]. For a more complete early survey of Steiner tree heuristic results, see [39]. The approximation guarantee for ST was subsequently improved to 11/6 [40], then to 16/9 [4], and so on [25, 31, 32, 41], to 1.598 [20]. Robins and Zelikovsky [34] then obtained an approximation guarantee of $1 + \frac{\ln 3}{2} \approx 1.55$ for ST, and a 1.28-approximation guarantee for quasi-bipartite graphs (graphs in which no two Steiner vertices are adjacent) and for complete graphs with edge costs in $\{0, 1\}$. However, in spite of this progress in obtaining approximation algorithms with better approximation guarantees for ST, the best known integrality gap for an LP relaxation of ST remained at 2. The fact that approximation guarantees and integrality gaps often are the same, suggested that perhaps there existed a different LP relaxation for ST that gives a smaller integrality gap. It remained a long-standing open problem to find an LP relaxation for ST that had an integrality gap less than 2 [27]. Könemann et al. presented an LP relaxation for ST with an integrality gap between $\frac{8}{7}$ and $\frac{2b+1}{b+1}$ [27] for b-quasi-bipartite graphs, but the integrality gap for ST on a general graph remained at 2. Very recently, Bykra et al. broke through this stalemate, presenting a hypergraph-based LP relaxation for ST that has an integrality gap of 1.55 [5] (an alternate proof and analysis is provided by [7]). Not only did this improve the integrality gap for ST to less than 2 for general graphs, but also it matched the best known approximation guarantee of 1.55 for ST. However, Bykra et al. also presented a 1.39-approximation algorithm for ST. It currently remains an open problem to find an LP relaxation for ST that has an integrality gap of 1.39.

In this paper, we investigate different ILP formulations for ST, and their corresponding LP relaxations. We consider finding LP relaxations for ST such that the LP relaxation, when restricted to MST, has an integrality gap of 1. Such an LP relaxation for ST may have an integrality gap closer to 1 and thus be able to be used to obtain stronger results for ST. Section 5.2 examines some known ILP formulations for ST, and the bounds on the integrality gap of their LP relaxations. In Section 5.3, we propose a new ILP formulation for ST that is based on the well-known undirected cut formulation. We give an example for which the integrality gap of the LP relaxation has a lower bound of 3/2; thereby, for this example, improving the integrality gap of the LP relaxation from being tight at 2 to having a lower bound of 3/2. However, we also show that when restricted to MST, the LP relaxation of this proposed formulation does not have an integrality gap of 1. Thus, the formulation does not look promising to use in obtaining improved results for ST. In Section 5.4, we propose a second new ILP formulation for ST, this one based on Steiner partitions. We show that, when restricted to MST, the LP relaxation of this proposed formulation, although similar to an LP for MST which has integrality gap 1, has a lower bound of 3/2 for its integrality gap (for MST). Additionally, a family of graphs is given that provides a lower bound of $8/5 = 1.6$ for the integrality gap of the LP relaxation of this proposed formulation.

5.2 Linear Programming Relaxations of ST

In this section, we state some known ILP formulations for ST, and the integrality gap of their respective LP relaxation. Let $G = (V, E)$ be a graph with non-negative edge costs $c \in \mathbb{R}^E$, and let $T \subseteq V$ be the terminal vertices. Let x_e represent the number of times each edge $e \in E$ is included in the solution. For any edge set $E' \subseteq E$ and $x \in \mathbb{R}^E$, let $x(E')$ denote the sum $\sum_{e \in E'} x_e$.

5.2.1 Undirected Cut Formulation

An integer linear programming formulation of the minimum Steiner tree problem is [2]:

$$\begin{aligned} & \min cx & & \qquad (5.1) \\ \text{subject to} \quad x(\delta(S)) & \geq 1, \text{ for all } S \subset V,\ S \text{ separates } T, \\ x_e & \in \{0,1\}, \text{ for all } e \in E. \end{aligned}$$

This is known as the *undirected cut formulation*. Notice that this formulation is precisely that of Jain's general network ILP [23] for the special case of ST (since for ST $f(S) \in \{0,1\}$, and we can remove the cut constraints $x(\delta(S)) \geq 0$, since they are automatically satisfied). Thus, the LP relaxation of the undirected cut formulation has an integrality gap of 2 [23]. Notice that in the LP relaxation of the undirected cut formulation, the constraints $x_e \leq 1$ for all $e \in E$ are redundant [38] and thus can be omitted.

For the special case of MST (i.e., $T = V$), the LP relaxation of the undirected cut formulation gives an integrality gap that is asymptotically close to 2. This can be seen as follows [23, 36]: Consider a cycle on n vertices, with all edges of edge cost 1. The optimal integral solution of MST on this graph is the solution consisting of $n-1$ edges (for a total edge cost of $n-1$). However, an optimal solution of the LP relaxation of the undirected cut formulation for the special case of MST consists of an x_e value of $1/2$ for each edge (for a total edge cost of $n/2$). See Figure 5.2. The ratio of the two solutions is $2 - 2/n$, which approaches 2 as n gets large. Thus, the integrality gap of the LP relaxation of the undirected cut formulation has a lower bound that is asymptotically close to 2. Thus, the integrality gap of 2 is tight.

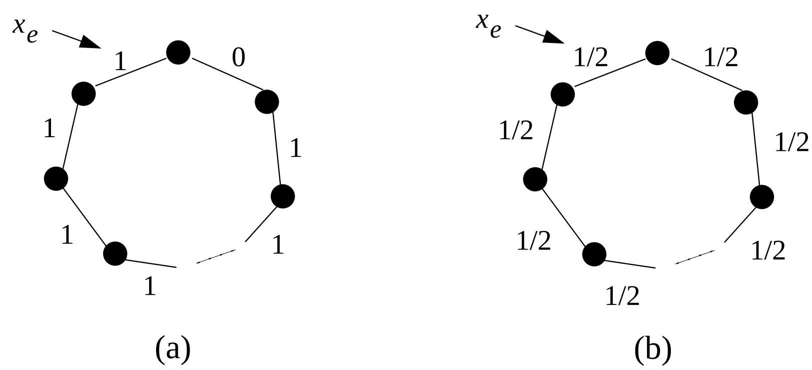

Figure 5.2: (a) An optimal solution of the undirected cut formulation, ILP(5.1), on the cycle, for the special case of MST. (b) An optimal solution of the LP relaxation of ILP(5.1) on the cycle, for the special case of MST. Edge costs are 1.

5.2.2 Bi-directed Cut Formulation

Another integer linear programming formulation for ST is the *bi-directed cut formulation*. This formulation is on a directed graph. To formulate the bi-directed cut relaxation for an undirected weighted graph $G = (V, E)$ with the set of terminal vertices $T \subseteq V$, replace each undirected edge $uv \in E$ by two directed edges (arcs) $\overrightarrow{uv}$ and $\overrightarrow{vu}$, each of cost c_{uv}. Denote the new directed graph by $\overrightarrow{G} = (V, \overrightarrow{E})$. Let the root $r \in T$ be an arbitrary vertex in T. A set $S \subset V$ is a *valid set* if it contains at least one terminal vertex and $V \backslash S$ contains the root r, i.e., if $|S \cap T| \geq 1$ and $r \notin S$. See Figure 5.3.

The following integer linear programming formulation solves the problem of finding a minimum cost collection of edges from $\overrightarrow{E}$ in such a way that each valid set has at least one arc directed out of it [33]:

$$\min cx \tag{5.2}$$

$$\begin{aligned} \text{subject to} \quad x(\delta(S)) &\geq 1, \text{ for all valid sets } S \subset V, \\ x_e &\in \{0, 1\}, \text{ for all } e \in \overrightarrow{E} \end{aligned}$$

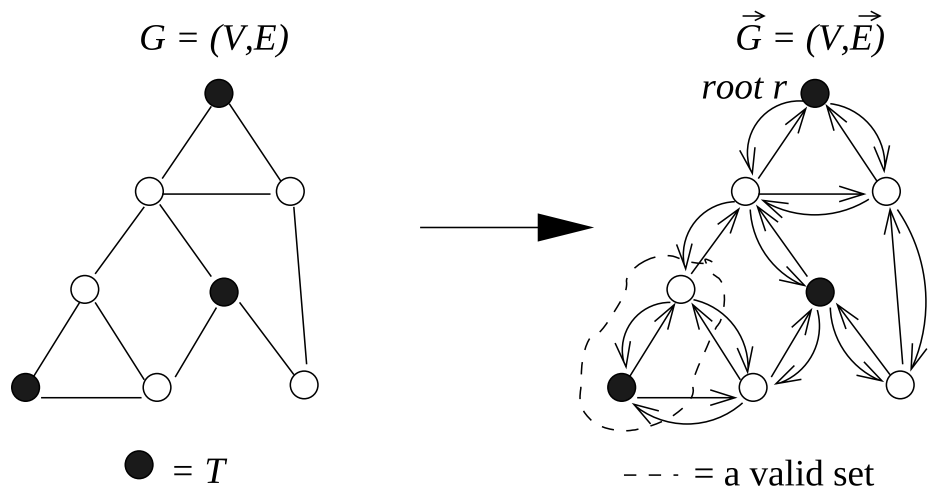

Figure 5.3: An example of transforming a graph to the proper form for the bi-directed cut relaxation.

An optimal solution to ILP(5.2) is a minimum cost directed Steiner in-tree rooted at r, i.e., a minimum cost directed Steiner tree that has no arcs directed out of r and every other terminal vertex has exactly one arc directed out of it [11, 37]. Removing the directions on the edges in the solution yields a feasible solution to ST. The constraints on valid sets $S \subset V$ are called *Steiner cut constraints*. They ensure that in every feasible solution there is a path from r to any other terminal vertex. The LP relaxation (i.e., with $x_e \geq 0$ for all $e \in \overrightarrow{E}$) of ILP(5.2) is called the *bidirected cut relaxation* for the metric Steiner tree problem. Notice that there is no need in the relaxation to explicitly have the constraints $x_e \leq 1$ for all $e \in \overrightarrow{E}$ [37].

The bi-directed cut relaxation was shown by Edmonds to be exact for the special case of MST [6]. In other words, the integrality gap is 1 for MST using this LP relaxation.

The integrality gap of the bi-directed cut relaxation is at most 2 [38]; however, the bi-directed cut relaxation is conjectured to have an integrality gap close to 1 [36, 37]. For this relaxation, the worst known examples, from Goemans and Skutella respectively, have an integrality ratio of $\frac{8}{7}$ [27, 38]. Skutella's example consists of the Fano graph with edge costs 1. An optimal Steiner tree solution has cost 10, while an optimal solution for the LP relaxation has x_e values of 1/4 on every edge, for a total cost of $\frac{35}{4}$. See Figure 5.4. Hence, Skutella's example has an 8/7 integrality ratio. Thus, we have that the integrality gap of the bi-directed cut relaxation is between 8/7 (≈ 1.14) and 2.

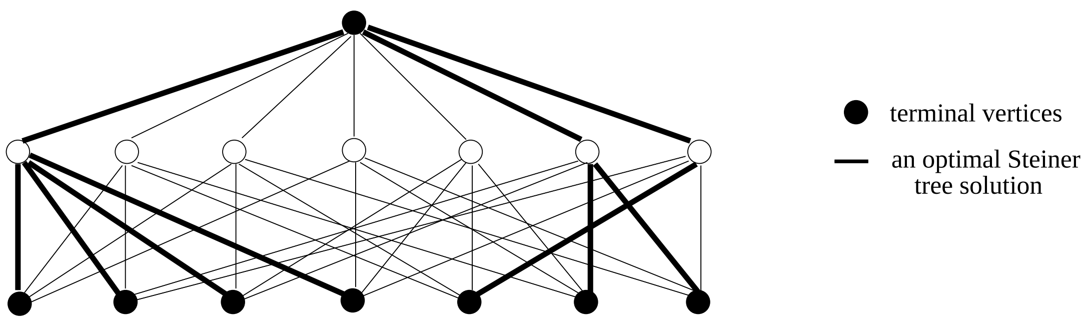

Figure 5.4: Skutella's example [27], which shows that the bi-directed cut formulation has an integrality gap of at least $\frac{8}{7}$. Edge costs are all 1.

For these reasons, and more (see [11]), the bi-directed cut relaxation is considered to be a better LP relaxation than the LP relaxation of the undirected cut formulation, and is used in many linear programming approaches for developing approximation algorithms with better (smaller) approximation guarantees for ST [1, 26]. For some of the more recent results for the bi-directed cut formulation, see [6, 33, 36].

5.2.3 ILP Formulation by Chopra and Rao

Chopra and Rao [11] show that the following is an integer linear programming formulation for ST:

$$
\begin{aligned}
& \min cx && && (5.3)\\
\text{subject to} \quad & x(\delta(S)) &\geq\ & 1, \text{ for all } S \subset V, S \text{ is a minimal Steiner cut of } G \\
& x_e &\geq\ & 0, \text{ for all } e \in E, \\
& x_e & & \text{integer}, \text{ for all } e \in E.
\end{aligned}
$$

Notice that this is similar to the undirected cut formulation, and, like the undirected cut formulation, the restriction of its LP relaxation to MST ($T = V$) does not have an integrality gap of 1.

A connected graph is *2-vertex connected* if any vertex can be removed without disconnecting the graph. Chopra and Rao [11] present some families of facet-defining inequalities for the Steiner tree polyhedron. In particular, they show the following:

Theorem 1 (Chopra and Rao, [11]) *The Steiner partition inequalities*

$$\sum_{e \in E_\Pi} x_e \geq k_\Pi - 1, \text{ for all Steiner partitions } \Pi$$

are valid for the Steiner tree polyhedron. Furthermore, these inequalities are facet defining for the Steiner tree polyhedron if and only if the shrunk graphs G/Π are 2-vertex connected and the corresponding induced subgraphs $G[V_i]$, $i = 1, 2, \ldots, k_\Pi$, are connected.

5.2.4 Other ILP Formulations

See [5, 7] for a hypergraph-based LP relaxation of ST. This LP relaxation has an integrality gap between 8/7 and 1.55 [5].

For more ILP formulations of ST, see [11, 12, 19, 27, 30]. For other facet defining inequalities for the Steiner tree polyhedron, see [18].

5.3 A New ILP formulation for ST

We investigate obtaining an ILP formulation for ST such that its LP relaxation has an integrality gap close to 1, and its LP relaxation has an integrality gap of 1 for MST.

In Section 5.2.1, we saw that ILP(5.1), the undirected cut formulation for ST, has an LP relaxation whose integrality gap is 2. We also saw that the LP relaxation of ILP(5.1) on the graph consisting of a cycle on n vertices and edge costs all 1 has an integrality gap asymptotically close to 2 for MST (thus making the integrality gap tight). This is due to the fact there as an optimal solution of the LP relaxation on that graph that has an x_e value of 1/2 for each edge. Cutting off this solution, i.e., adding a constraint(s) such that this solution is no longer feasible for the LP relaxation, might yield an LP relaxation whose restriction to MST has integrality gap less than 2 and closer to 1. Notice that $\frac{1}{2}n \not\geq n-1$. Thus, the constraint $\sum_{e \in E} x_e \geq |T|-1$ will cut off the solution (recall that T is the set of terminal vertices, and $T = V$ for MST). We thus propose a new ILP for ST by adding the constraint $\sum_{e \in E} x_e \geq |T|-1$ to ILP(5.1):

$$\begin{aligned} & \min cx && (5.4) \\ \text{subject to} \quad x(\delta(S)) &\geq 1, \text{ for all } S \subset V,\ S \text{ separates } T, \\ \sum_{e \in E} x_e &\geq |T|-1, \\ x_e &\in \{0,1\}, \text{ for all } e \in E. \end{aligned}$$

For the special case MST (i.e., $T = V$), the LP relaxation of ILP(5.4) becomes:

$$\begin{aligned} & \min cx && (5.5) \\ \text{subject to} \quad x(\delta(S)) &\geq 1, \text{ for all } S \subset V, \\ \sum_{e \in E} x_e &\geq n-1, \\ 0 \leq x_e &\leq 1, \text{ for all } e \in E. \end{aligned}$$

5.3.1 Integrality Gap of LP(5.5)

We investigate the strength of this LP relaxation for MST by comparing it with the following well-known LP formulation for MST that has integrality gap 1 [15]:

$$\begin{aligned} & & \min cx & & & \quad (5.6) \\ \text{subject to} & & x(\gamma(S)) & \leq |S|-1, \text{ for all } S \subset V, \\ & & \sum_{e\in E} x_e & = n-1, \\ & & x_e & \geq 0, \text{ for all } e \in E \end{aligned}$$

If the LPs are equivalent (i.e., their feasible regions are identical), then LP(5.5) has integrality gap 1 for MST, and thus the LP relaxation of ILP(5.4) might be a promising LP formulation to consider for ST.

Lemma 2 *All feasible solutions of LP(5.6) are also feasible solutions of LP(5.5).*

Proof: Suppose we have a feasible solution x of LP(5.6), i.e., x satisfies the constraints of LP(5.6). Then, for any $S \subset V$, we have that

$$\begin{aligned} x(\gamma(S)) & \leq |S|-1, \text{ and} \\ x(\gamma(V\backslash S)) & \leq |V\backslash S|-1. \end{aligned}$$

Adding these inequalities we get

$$x(\gamma(S)) + x(\gamma(V\backslash S)) \leq |S| + |V\backslash S| - 2 = n-2. \quad (5.7)$$

From properties of cuts, $\gamma(V) = E = \gamma(S) \cup \gamma(V\backslash S) \cup \delta(S)$, thus $x(\gamma(V)) = x(\gamma(S)) + x(\gamma(V\backslash S)) + x(\delta(S))$. Rearranging, we get

$$x(\delta(S)) = x(\gamma(V)) - x(\gamma(S)) - x(\gamma(V\backslash S)).$$

From the constraints of LP(5.6), $x(\gamma(V)) = n-1$ and thus,

$$\begin{aligned} x(\delta(S)) & = (n-1) - x(\gamma(S)) - x(\gamma(V\backslash S)) \\ x(\delta(S)) - n + 1 & = -x(\gamma(S)) - x(\gamma(V\backslash S)). \quad (5.8) \end{aligned}$$

Multiplying (5.7) by -1, yields: $-x(\gamma(S)) - x(\gamma(V \setminus S)) \geq -n+2$. Inserting this into (5.8), we obtain

$$\begin{aligned} x(\delta(S)) - n + 1 &\geq -n + 2 \\ \text{i.e.,}\;\; x(\delta(S)) &\geq 1. \end{aligned}$$

In addition, notice that for subsets $S \subset V$ of size 2, $x(\gamma(S)) \leq 1$, which implies that $x_e \leq 1$ for all $e \in E$. Therefore, x is also a feasible solution of LP(5.5). Thus, we have that the solution space of LP(5.6) is contained in the solution space of LP(5.5).

■

Lemma 3 *There exist feasible solutions of LP(5.5) that are not feasible solutions of LP(5.6).*

Proof: Consider the triangular graph shown in Figure 5.5. In this graph, costs on the edges are 1 and the x_e values of a feasible solution of LP(5.5) are the values shown on the edges. Note that such an x satisfies the constraints $\sum_{e \in E} x_e = n - 1$, as well as the constraints $x(\delta(S)) \geq 1$ for all $S \subset V$, and thus is a feasible solution of LP(5.5). However, such an x does not satisfies the constraints $x(\gamma(S)) \leq |S| - 1$ for all $S \subset V$, and thus it is not a feasible solution of LP(5.6). To see this, let S' be the set of vertices of one of the "corner triangles" of the example in Figure 5.5. Then $x(\gamma(S')) = 2\frac{1}{6} \nleq |S'| - 1 = 2$. ■

Thus, from Lemmas 2 and 3, LP(5.5) and LP(5.6) are not equivalent. However, as a minimization problem, the optimal extreme points of the underlying polytope of LP(5.5) may still be integer (i.e., the lower part of the feasible region of LP(5.5) may match that of LP(5.6) and thus the optimal solutions of LP(5.5) may be precisely the optimal solutions of LP(5.6)), and thus LP(5.5) may still have an integrality gap of 1 for MST. That this is not the case can be seen by the following example: Consider the triangular graph mentioned in the proof of Lemma 3 and let the edges of the three small triangles have edge cost 1, and the remaining three edges have edge cost

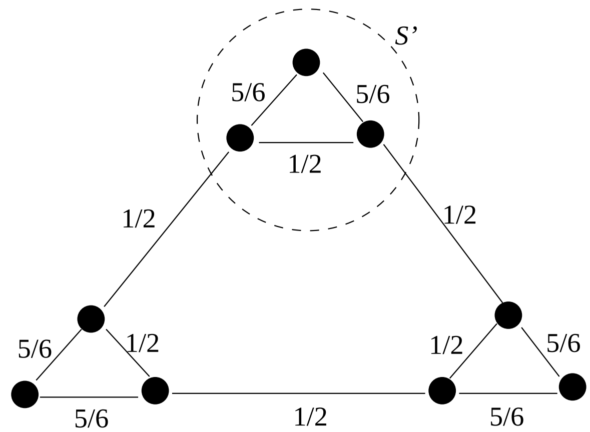

Figure 5.5: Example showing that LP(5.5) is not equivalent to LP(5.6). Edge costs are all 1.

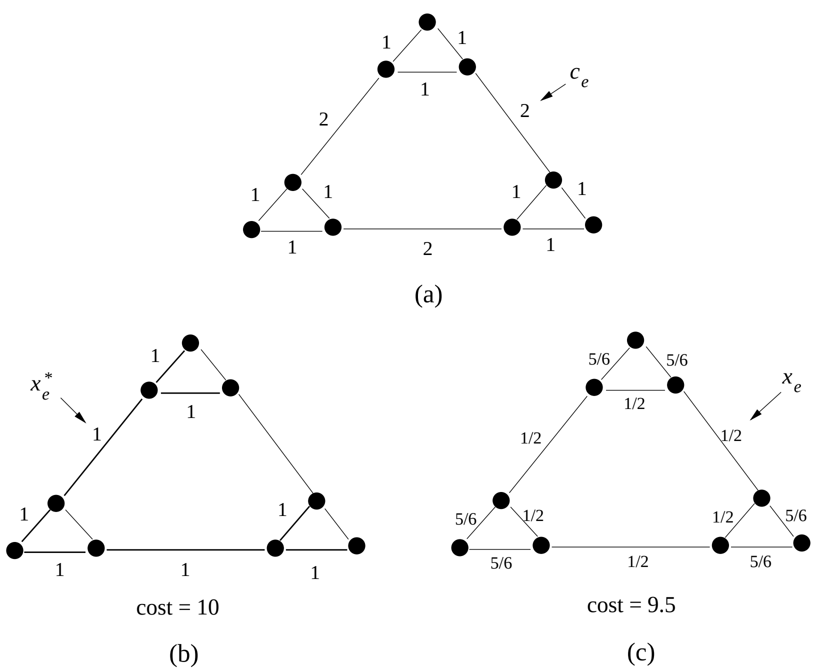

Figure 5.6: Example that LP(5.5) does not have an integrality gap of 1: (a) the edge costs on the graph, (b) an optimal MST solution, x^*, (c) a feasible LP solution, x, with smaller total cost.

2. See Figure 5.6(a). An optimal MST solution uses six edges of cost 1 and two edges of cost 2, for a total cost of 10. See Figure 5.6(b). The feasible solution of LP(5.5) that we just examined has, for these edge costs, a total cost of 9.5. See Figure 5.6(c). Therefore, the cost of an optimal solution to LP(5.5) is no more than 9.5, and thus the integrality ratio is at least $\frac{10}{9.5} = \frac{20}{19}$. Since we have an example where the integrality ratio is greater than 1, therefore LP(5.5) does not have an integrality gap of 1 for MST.

5.3.2 A Lower Bound of 3/2 for the Integrality Gap of the LP Relaxation of ILP(5.4) for ST

Consider the wheel graph, consisting of three "rim" vertices and one "center" vertex. See Figure 5.7. Costs on the rim edges are $2-\beta$, $0 < \beta < 1$, and costs on the "spoke" edges are 1. The rim vertices are terminal vertices.

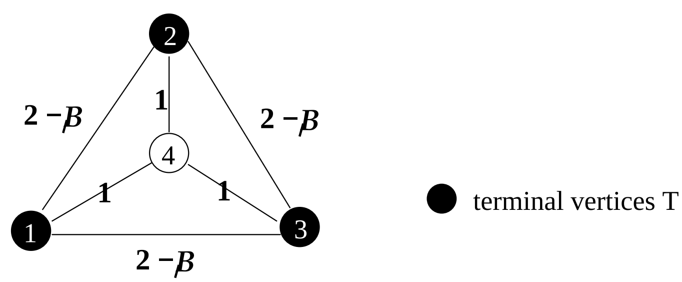

Figure 5.7: A wheel example for which the ratio of the costs of the optimal solutions of ILP(5.4) and its LP relaxation asymptotically approaches 3/2. Edge costs are shown on edges.

For this wheel example, the relaxation of ILP(5.4) is (since $|T| - 1 = 2$):

$$\begin{aligned} & \min cx && && (5.9) \\ \text{subject to} \quad & x(\delta(S)) && \geq 1, && \forall\, S \subset V \text{ that separates } T, \\ & \sum_{e \in E} x_e && \geq 2, && \\ & 0 \leq x_e && \leq 1, && \forall\, e \in E. \end{aligned}$$

The separating sets S are the vertex sets {1}, {2}, {3}, {1,2}, {1,3}, {2,3}, and their complements {1,2,4}, {1,3,4}, {2,3,4}, {1,4}, {2,4}, {3,4} (which give the same constraints). Thus, specifically, the relaxation of ILP(5.4) on this wheel example is:

$$\min\ (2-\beta)x_{12} + (2-\beta)x_{13} + x_{14} + (2-\beta)x_{23} + x_{24} + x_{34}$$

$$\begin{aligned} \text{subject to} \quad x_{12} + x_{13} + x_{14} &\geq 1 \\ x_{12} + x_{23} + x_{24} &\geq 1 \\ x_{13} + x_{23} + x_{34} &\geq 1 \\ x_{13} + x_{14} + x_{23} + x_{24} &\geq 1 \\ x_{12} + x_{14} + x_{23} + x_{34} &\geq 1 \\ x_{12} + x_{24} + x_{13} + x_{34} &\geq 1 \\ x_{12} + x_{13} + x_{14} + x_{23} + x_{24} + x_{34} &\geq 2 \\ 0 \leq x_{12} &\leq 1 \\ 0 \leq x_{13} &\leq 1 \\ 0 \leq x_{14} &\leq 1 \\ 0 \leq x_{23} &\leq 1 \\ 0 \leq x_{24} &\leq 1 \\ 0 \leq x_{34} &\leq 1 \end{aligned}$$

An optimal integer solution, x', to ILP(5.4) for the wheel has $x'_e = 0$ for the rim edges and $x'_e = 1$ for the spoke edges (see Figure 5.8a). The cost of this optimal ILP solution is 3. Using the dual of the above LP, an optimal LP solution x^* for the wheel is $x^*_{12} = x^*_{13} = x^*_{24} = x^*_{34} = \frac{1}{2}$ and $x^*_{14} = x^*_{23} = 0$ (see Figure 5.8b). Thus, the optimal value for the LP relaxation is $(2 - \beta) + 1 = 3 - \beta$. The ratio of the optimal ILP and optimal LP relaxation of ILP(5.4) for this wheel is $\frac{3}{3-\beta}$, which asymptotically approaches 3/2 as β approaches 1.

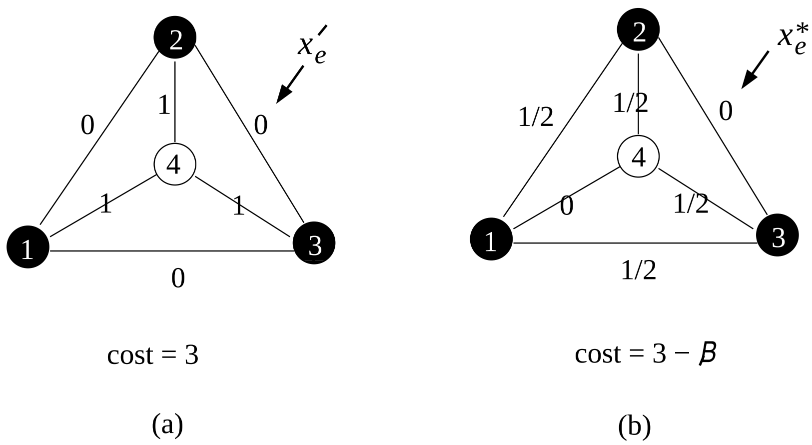

Figure 5.8: (a) An optimal solution of ILP(5.4) for the wheel example in Figure 5.7, and (b) a basic solution of the corresponding LP relaxation, LP(5.9).

Therefore, there is a lower bound of 3/2 on the integrality gap of the LP relaxation of ILP(5.4).

5.4 Another New ILP formulation for ST

In Section 5.3, we presented an ILP formulation of ST whose LP relaxation did not allow solutions consisting of x_e values of 1/2 on cycles through just the terminal vertices (and x_e values of 0 everywhere else). However, that LP relaxation still permits solutions consisting of x_e values of 1/2 on cycles through a combination of terminal and Steiner vertices. See Figure 5.8b. We can obtain a stronger LP relaxation of ST by cutting off these solutions as well. Adding to ILP(5.4) the constraint that the sum of the x_e values must be at least $|T|$ plus the number of Steiner vertices used, minus 1, would accomplish this; however, this is difficult to state as a constraint. On the other hand, notice that the Steiner partition inequalities in Theorem 1 for Steiner partitions with $k_\Pi \geq 3$ also cut off solutions consisting solely of a cycle of x_e values of 1/2. See, for example, Figure 5.9. Thus, we investigate using Steiner partition inequalities in an LP formulation for ST.

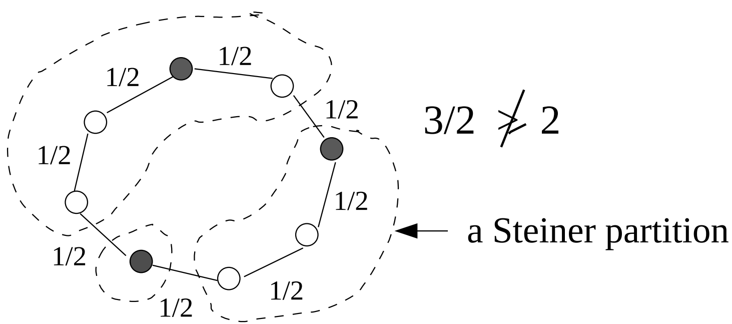

Figure 5.9: A Steiner partition inequality of size 3 cutting off a solution consisting of a cycle of edges with x_e values 1/2. Edges shown have x_e value 1/2. All other edges have x_e value 0.

Notice that, for Steiner partitions of size 2, the Steiner partition inequalities reduce to: $x(\delta(S)) \geq 1$ for all $S \subset V$ such that S separates T. These are precisely the cut constraints of the undirected cut formulation of ST, ILP(5.1). Since, by Theorem 1, the Steiner partition inequalities are valid for ST, the following integer linear programming formulation is an ILP for ST:

$$\begin{aligned} & \min cx && && (5.10)\\ \text{subject to} \quad & \sum_{e \in E_\Pi} x_e && \geq k_\Pi - 1, && \text{for all Steiner partitions } \Pi \text{ (of } G\text{) of} \\ & && && \text{size } k_\Pi \in \{2,3\} \text{ and such that each} \\ & && && G[V_i] \text{ is connected, } i = 1, 2, \ldots, k_\Pi, \\ & x_e && \in \{0,1\}, && \text{for all } e \in E. \end{aligned}$$

For the special case of MST, the LP relaxation of ILP(5.10) is the following:

$$\begin{aligned} & \min cx && && (5.11)\\ \text{subject to} \quad & \sum_{e \in E_P} x_e && \geq k_P - 1, && \text{for all partitions } P \text{ (of } G\text{) of size} \\ & && && k_P \in \{2,3\}, \text{ and such that each} \\ & && && G[V_i] \text{ is connected, } i = 1, 2, \ldots, k_P, \\ & x_e && \geq 0, && \forall\, e \in E. \end{aligned}$$

This is similar to, but broader than, Chopra's LP relaxation of MST, which has integrality gap 1 [10]:

$$\begin{aligned} & \min cx \\ \text{subject to} \quad & \sum_{e \in E_P} x_e \geq k_P - 1, \quad \forall \text{ feasible partitions } P \text{ of } G, \\ & x_e \geq 0, \quad \forall\, e \in E, \end{aligned}$$

where a partition $P = (\hat{V}_1, \hat{V}_2, \ldots, \hat{V}_{k_P})$ of V is *feasible* if each induced subgraph $G[\hat{V}_i]$ is connected. Notice that Chopra's LP has constraints for all feasible partitions of G, whereas LP(5.11) has constraints for only feasible partitions of size 2 or 3. Thus, although similar to Chopra's LP, LP(5.11) does not necessarily have an integrality gap of 1 for the LP relaxation of MST.

5.4.1 A Lower Bound for the Integrality Gap of the LP Relaxation of ILP(5.10)

Cycle Example

Let G to be the cycle on n vertices, with all edges of cost 1 and $T = V$. The optimal solution of ILP(5.10) (in this case, the optimal MST solution) consists of $n-1$ edges. An optimal solution to LP(5.11) has x_e values of $2/3$ on each edge. See Figure 5.10. The ratio of the two solutions is $\frac{n-1}{2/3n} = \frac{3(n-1)}{2n} = \frac{3}{2} - \frac{3}{n}$, which approaches $\frac{3}{2}$ as n gets large.

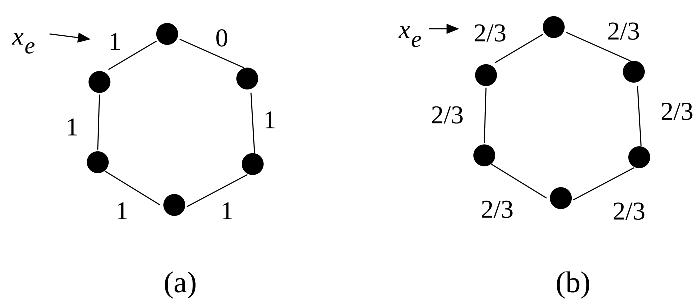

Figure 5.10: An example that has a ratio of 3/2 for the ratio of the optimal cost of ILP(5.10) to the optimal cost of the LP relaxation of ILP(5.10), restricted to MST: (a) an optimal solution of ILP(5.10), (b) an optimal solution of LP(5.11).

This gives a lower bound of 3/2 on both the integrality gap of the LP relaxation of ILP(5.10) and the integrality gap of LP(5.11). It also shows that, while solutions that are cycles consisting of x_e values of $1/2$ are cut off by our formulation, solutions that are cycles consisting of x_e values of $2/3$ are not cut off. As we see in the next example (Section 5.4.1), there is a larger lower bound for the integrality gap of the LP relaxation of ILP(5.10).

Odd-hole Example

An odd-hole graph $H_k = (V_k, E_k)$ is a graph that has $2k$ vertices ($k \in \mathbb{N}_{>0}$), with terminal vertices $T_k = \{t_1, t_2, \dots, t_k\}$ and Steiner vertices $V_k \backslash T_k = \{v_1, v_2, \dots, v_k\}$ placed alternating in a cycle. The edge set E_k consists of the "rim" edges connecting the alternating terminal and Steiner vertices in an "exterior" cycle (i.e., the edges of the form $e_{t_i v_i}$ and $e_{t_i v_{i+1}}$, $i = 1, 2, \dots, k$, where $v_{k+1} := v_1$), and the edges that join just the Steiner nodes in an "interior" cycle (i.e., the edges of the form $e_{t_i, t_{i+1}}$, $i = 1, 2, \dots, k$, where $t_{k+1} := t_1$).

Let $P_{ST}(G)$ be the Steiner tree polyhedron on graph G. Chopra and Rao [11] present a family of odd-hole inequalities that is facet-defining for the Steiner tree polyhedron on the odd-hole graphs, H_k.

Theorem 4 (Chopra and Rao, [11]) *Given the odd-hole graph H_k, the odd-hole inequality*

$$\sum_{e \in E_k} x_e \geq 2(k-1) \tag{5.12}$$

is valid for $P_{ST}(H_k)$ if $k \geq 3$. Furthermore, the odd-hole inequality is facet defining for $P_{ST}(H_k)$ if k is odd and $k \geq 3$.

We use these odd-hole facet-defining inequalities for the odd-hole graph to create an example on which to investigate the LP relaxation of ILP(5.10).

In general, to form an example from a facet-defining inequality $ax \geq b$ for a polytope on G, $\mathcal{P}(G)$, we do the following: For each edge e of G, we take the coefficient of x_e in the inequality and set it to be the cost of e, i.e. set $c_e = a_e$. Then ax is precisely the objective function of the LP. Since the inequality is valid for $\mathcal{P}(G)$, we know that the value of the objective function is at least b. Thus, if we find an integer solution with value b, then we know it is an optimal solution. Since the coefficients of the odd-hole inequality (5.12) are all 1, set all the edge costs in the odd-hole graph to be 1. We look at the LP relaxation of ILP(5.10) on the odd-hole graph H_k with edge costs 1, for $k = 3$, $k = 5$, and as k gets large.

Odd-hole with $k = 3$

Consider the odd-hole graph H_3 with edge costs all 1 (see Figure 5.11). Appendix A contains the formulation of the LP relaxation of ILP(5.10) on H_3.

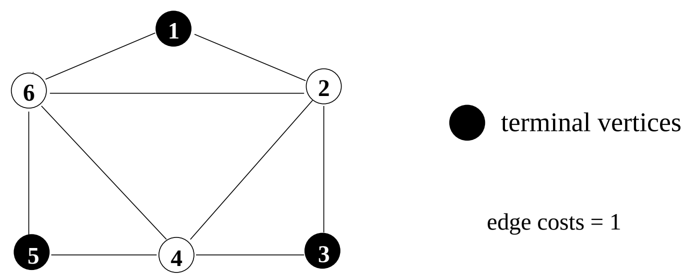

Figure 5.11: The odd-hole graph $H_k = 3$. Edge costs are all 1.

Using the dual, an optimal solution for the LP relaxation of ILP(5.10) on H_3 consists of x_e values $\frac{1}{2}$ on the "outside" edges, and x_e values $\frac{1}{4}$ on the "inside" edges, for a total cost of $\frac{15}{4}$, or $3\frac{3}{4}$ (see Figure 5.12). Notice that the tight constraints are precisely the inequalities whose corresponding Steiner partition has a part consisting of just a terminal node. An optimal solution ILP(5.10) on H_3 consists of a path, starting from a terminal vertex, of 4 outside edges. This gives an integrality ratio of $\frac{16}{15} = 1.0\bar{6}$.

It can be easily seen that for this odd-hole graph H_3, edge costs all 1 yields the largest integrality ratio. It is interesting to note that, for this example, the addition in the LP relaxation of ILP(5.10) of constraints corresponding to Steiner partitions of size 2 or 3 for which the induced subgraphs $G[V_i]$, $i \in \{1, 2, 3\}$, are not necessarily connected, does not change the optimal fractional solution[3].

[3]Indeed, running the CPLEX Interactive Optimizer on the LP formulation with constraints corresponding to all the Steiner partitions of size 2 or 3 returned the same LP solution as we obtained

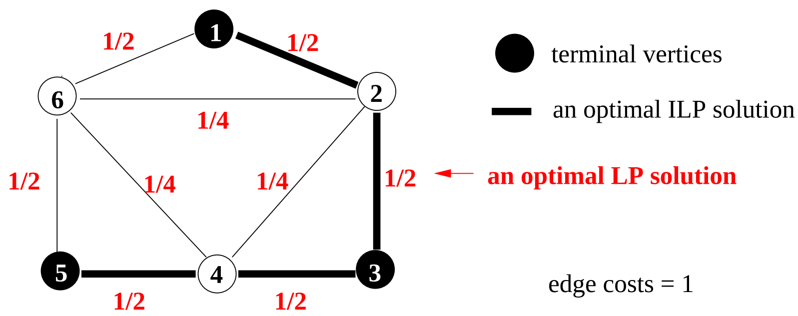

Figure 5.12: Optimal ILP and LP solutions for ILP(5.10) and its LP relaxation, respectively, on the odd hole graph H_3. Edge costs are all 1.

Odd-hole with $k = 5$

Consider the odd-hole graph H_5 with edge costs all 1 (see Figure 5.13). Appendix B contains the formulation of the LP relaxation of ILP(5.10) on H_5.

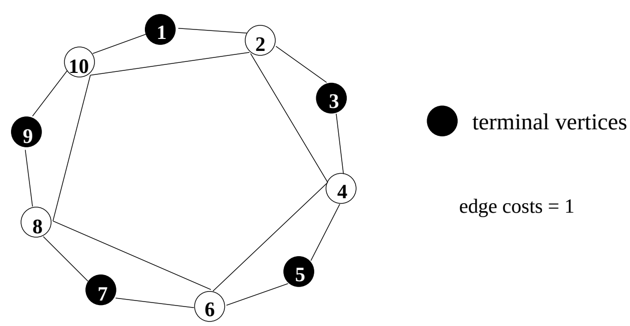

Figure 5.13: The odd-hole graph H_k. Edge costs are all 1.

Using the dual, an optimal solution for the LP relaxation of ILP(5.10) on H_5 consists of x_e values $\frac{1}{2}$ on the outside edges, and x_e values $\frac{1}{4}$ on the inside edges, for a total cost of $\frac{25}{4}$, or $6\frac{1}{4}$ (see Figure 5.14). The tight constraints are precisely the inequalities whose corresponding Steiner partition has a part consisting of just a

for the LP relaxation of ILP(5.10).

terminal node. An optimal solution ILP(5.10) on H_5 consists of a path, starting from a terminal vertex, of 8 outside edges. This gives an integrality ratio of $\frac{32}{25} = 1.28$.

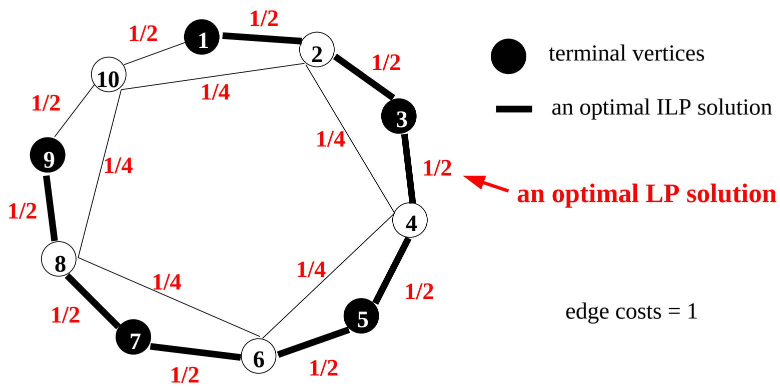

Figure 5.14: Optimal ILP and LP solutions for ILP(5.10) and its LP relaxation, respectively, on the odd hole graph H_5. Edge costs are all 1.

Notice that this optimal LP solution is of the same form as that for the odd-hole graph H_3. For the odd-hole graph H_3, edge costs of all 1 gives the largest integrality ratio. The same holds for the odd-hole graph H_5.

Odd-hole with large k

Generalizing the results for the odd-hole graphs H_3 and H_5, we obtain that for the odd-hole family of graphs H_k, $k \geq 3$, with edge costs 1, an optimal basic solution for the LP relaxation of ILP(5.10) on H_k consists of x_e values $\frac{1}{2}$ on the outside edges, and x_e values $\frac{1}{4}$ on the inside edges, for a total cost of $2k(\frac{1}{2}) + k(\frac{1}{4}) = k + k(\frac{1}{4}) = \frac{5}{4}k$. The tight constraints are precisely the inequalities whose corresponding Steiner partition has a part consisting of just a terminal node. An optimal solution of ILP(5.10) on H_k consists of a path, starting from a terminal vertex, of $2k-2$ outside edges, for a total cost of $2k-2$. Thus, the integrality ratio for the LP relaxation of ILP(5.10) on this odd-hole family of graphs for $k \geq 3$ is $\frac{2k-2}{\frac{5k}{4}} = (2k-2)\frac{4}{5k} = \frac{8k-8}{5k} = \frac{8}{5} - \frac{8}{5k}$, which approaches $\frac{8}{5} = 1.6$ as k gets large.

Thus, the integrality ratio for the LP relaxation of ILP(5.10) on for the odd-hole family of graphs H_k, $k \geq 3$, using just the facet-defining Steiner partitions of size 2 and 3, asymptotically approaches $\frac{8}{5} = 1.6$.

Therefore, we have that the lower bound on the integrality gap for the LP relaxation of ILP(5.10) asymptotically approaches $\frac{8}{5}$.

5.5 Discussion

Notice that there are Steiner partitions of size 4 (such that the induced subgraphs on the parts of the Steiner partition are connected) that cut off the optimal LP solution to the LP relaxation of ILP(5.10). For example, for the odd-hole graph H_5, the Steiner partition $(\{1,2\},\{3,4,5,6\},\{7\},\{8,9,10\})$ is a Steiner partition of size 4 whose induced subgraphs on the parts are connected, and whose corresponding constraint cuts off the optimal LP solution we saw in Section 5.4.1, since $\sum_{e \in E_\Pi} x_e = 4(\frac{1}{2}) + 3(\frac{1}{4}) = 2\frac{3}{4} \not\geq 3$. See Figure 5.15.

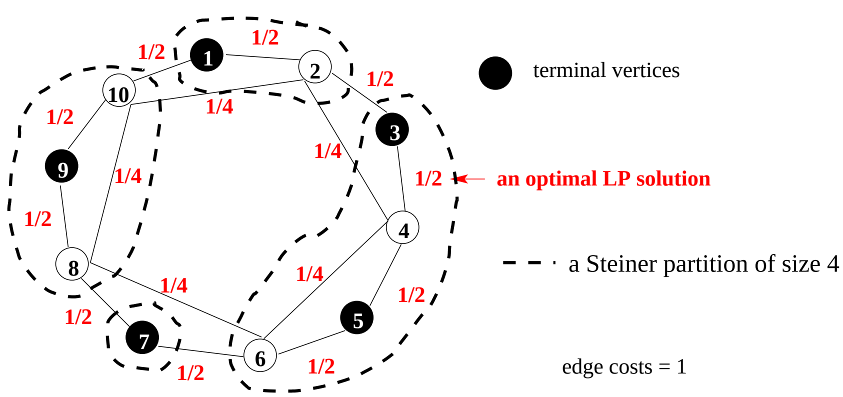

Figure 5.15: A Steiner partition of size 4 whose induced subgraphs on the parts are connected, which cuts off the optimal LP solution to the LP relaxation of ILP(5.10) on the odd-hole graph H_5.

Notice that there are also Steiner partitions of size 4 that cut off the cycle of

x_e values of 2/3 in the optimal solution of LP(5.11) for the cycle example in Section 5.4.1. Clearly, adding in constraints corresponding to Steiner partitions of size 4 (such that the induced subgraphs on the parts of the Steiner partition are connected), can only possibly make the integrality gap of the LP relaxation of ILP(5.10) smaller. Further investigation could be done to see whether adding inequalities corresponding to Steiner partitions of size 4 or greater (such that the induced subgraphs on the parts of the Steiner partition are connected) to ILP(5.10) would greatly change the 8/5 lower bound on the integrality gap, or whether there is a different example that gives a larger lower bound on the integrality gap.

In this paper, we have proposed and investigated two ILP formulations of the Steiner tree problem. The first proposed formulation cuts off some solutions of the undirected cut formulation, thus, for one example, reducing the integrality gap from being tight at 2 to having a lower bound of 3/2. However, for the special case of MST the LP relaxation of this proposed formulation does not have an integrality gap of 1, and thus the formulation does not look promising to use in obtaining improved results for Steiner tree.

The second proposed ILP formulation of Steiner tree is based on Steiner partitions. For the special case of MST, the LP relaxation once again does not have an integrality gap of 1, but is similar to an LP relaxation of MST which does have an integrality gap of 1. A family of graphs was presented that provides a lower bound of $8/5 = 1.6$ for the integrality gap of the LP relaxation of this proposed formulation. The hypergraph-based LP relaxation of ST by Byrka et al. [5], with an integrality gap of 1.55, remains the current LP relaxation with the lowest known integrality gap.

It remains an open problem to determine whether or not the integrality gap of the bi-directed cut relaxation is less than 2 [5]. It also remains an open problem to find an LP relaxation for ST that has an integrality gap of 1.39 (thus matching the current best known approximation guarantee of 1.39 for ST [5]).

5.6 Acknowledgments

This research was partially supported by grants from the National Sciences and Engineering Research Council of Canada. Thanks to Joseph Cheriyan for directing the author to Byrka et al.'s recent approximation guarantee and integrality gap for a LP relaxation of ST [5], and to Deeparnab Chakrabarty for sending their alternate proof of that integrality gap [7].

Bibliography

[1] E. Althaus, T. Polzin, S.V. Daneshmand, Improving Linear Programming Approaches for the Steiner Tree Problem, WEA 2003 LNCS 2647 (2003) 114.

[2] Y.P. Aneja, An Integer Linear Programming Approach to the Steiner Problem in Graphs, Networks 10 (1980) 167178.

[3] A. Balakrishnan, N. Patel, Problem Reduction Methods and a Tree Generation Algorithm for the Steiner Network Problem, Networks 17 (1987) 65-85.

[4] P. Berman, V. Ramaiyer, Improved Approximations for the Steiner Tree Problem, Proc. 3rd ACM-SIAM Symposium on Discrete Algorithms (1992) 325-334.

[5] J. Byrka, F. Grandoni, T. Rothvoβ, L. Sanità, An Improved LP-based Approximation for Steiner Tree, STOC 2010, Cambridge, Massachusetts, USA (2010).

[6] D. Chakrabarty, N. Devanur, V. Vazirani, New Geometry-inspired Relaxations and Algorithms for the Metric Steiner Tree Problem, Math Programming Series A (2009). Preliminary version appeared in the Proceedings of the 13th Conference on Integer Programming and Combinatorial Optimization (IPCO) (2008).

[7] D. Chakrabarty, J. Könemann, D. Pritchard, Integrality Gap of the Hypergraphic Relaxation of Steiner Trees: A Short Proof of a 1.55 Upper Bound, Operations Research Letters 38 (2010) 567570.

[8] J. Cheriyan, R. Ravi, Lecture Notes on Approximation Algorithms for Network Problems, http://www.math.uwaterloo.ca/~jcheriya/lecnotes.html. Accessed December 2011.

[9] M. Chlebík, J. Chlebíková, The Steiner Tree Problem on Graphs: Inapproximability Results, Theoretical Computer Science 406 (2008) 207214. Preliminary version: Approximation hardness of the Steiner tree problem on graphs, Proceedings, Scandinavian Workshop on Algorithm Theory (2002) 170179.

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

[11] S. Chopra, M.R. Rao, The Steiner Tree Problem I: Formulations, Compositions and Extension of Facets, Mathematical Programming 64 (1994) 209-229.

[12] S. Chopra, M.R. Rao, The Steiner Tree Problem II: Properties and Classes of Facets, Mathematical Programming 64 (1994) 231-246.

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

[14] S.E. Dreyfus, R.A. Wagner, The Steiner Problem in Graphs, Networks 1 (1972) 195-207.

[15] J. Edmonds, Matroids and the Greedy Algorithm, Mathematical Programming 1 (1971) 127-136.

[16] 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).

[17] E.N. Gilbert and H.O. Pollak, Steiner Minimal Trees, SIAM J. Appl. Math. 16 (1968) 1-29.

[18] M.X. Goemans, The Steiner Tree Polytope and Related Polyhedra, Mathematical Programming 63 (1994) 157-182.

[19] M.X. Goemans, Y. Myung, A Catalog of Steiner Tree Formulations, Networks 23 (1993) 19-28.

[20] S. Hougardy, H. J. Prömel, A 1.598 Approximation Algorithm for the Steiner Problem in Graphs, Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, Philadelphia, ACM, New York (1999) 448453.

[21] F.K. Hwang, On Steiner Minimal Trees with Rectilinear Distance, SIAM J. Appl. Math. 30 (1976) 104-114.

[22] F.K. Hwang, D.S. Richards, P. Winter, The Steiner Tree Problem, North-Holland, Amsterdam (1992).

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

[24] R.M. Karp, Reducibility Among Combinatorial Problems, in: Miller and Thatcher (Eds.), Complexity of Computer Computations, Plenum Press, New York (1972) 85-103.

[25] M. Karpinski, A. Zelikovsky, New Approximation Algorithms for the Steiner Tree Problem, Journal of Combinatorial Optimization 1 (1997) 4765.

[26] T. Koch, A. Martin, Solving Steiner Tree Problems in Graphs to Optimality, Networks 32 (1998) 207232.

[27] J. Könemann, D. Pritchard, K. Tan, A Partition-based Relaxation for Steiner Trees, Math. Programming Series A (2009).

[28] L. Kou, G. Markowsky, L. Berman, A fast Algorithm for Steiner trees, Acta Inform. 15 (1981) 141-145.

[29] L. Nastansky, S.M. Selkow, N.F. Stewart, Cost-minimal Trees in Directed Acyclic Graphs, Zeitschrijt fur Operations Res. 18 (1974) 59-67.

[30] T. Polzin, S.V. Daneshmand, A Comparison of Steiner Tree Relaxations, Discrete Applied Mathematics 112 (2001) 241-261.

[31] H.J. Prömel, A. Steger, A New Approximation Algorithm for the Steiner Tree Problem with Performance Ratio 5/3, Journal of Algorithms 36 (2000) 89-101.

[32] H.J. Prömel, A. Steger, RNC-approximation Algorithms for the Steiner Problem, Proceedings of the 14th Annual Symposium on Theoretical Aspects of Computer Science, Springer, Berlin (1997) 559570.

[33] S. Rajagopalan, V. Vazirani, On the Bidirected Cut Relaxation for the Metric Steiner Tree Problem, Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms (1999).

[34] G. Robins, A. Zelikovsky, Tighter Bounds for Graph Steiner Tree Approximation, SIAM J. Discrete Math 19 No.1 (2005) 122-134.

[35] H. Takahashi and A. Matsuyama, An Approximate Solution for the Steiner Problem in Graphs, Math, Japonica 24 (1980) 573-577.

[36] V.V. Vazirani, Recent Results on Approximating the Steiner Tree Problem and its Generalizations, Theoretical Computer Science 235 (2000) 205-216.

[37] V.V. Vazirani, The Steiner Tree Problem and its Generalizations, Lecture Notes in Computer Science 1444 (1008) 33-38.

[38] V.V. Vazirani, Approximation Algorithms, Springer-Verlag Berlin Heidelberg New York (2003).

[39] P. Winter, Steiner Problem in Networks: A Survey, Networks 17 (1987) 129-167.

[40] A. Z. Zelikovsky, The 11/6 Approximation Algorithm for the Steiner Problem on Networks, Algorithmica 9 (1993) 463-470.

[41] A. Zelikovsky, Better Approximation Bounds for the Network and Euclidean Steiner Tree Problems, Tech. report CS-96-06, University of Virginia, Charlottesville, VA (1996).

Appendix A

LP of Odd-hole Graph H_3

The LP formulation for the LP relaxation of ILP(5.10) on the odd-hole graph H_3, in an lp form that can be submitted to CPLEX, is the following. Backslashes are comments in CPLEX, and the number in brackets at the end of each line indicates the partition giving rise to that constraint.

```
Minimize
obj: x12 + x23 + x34 + x45 + x56 + x16 + x26 + x24 + x46
Subject To
\ steiner partition facets, size 2
c1: x12 + x16 >= 1      \(1)
c2: x23 + x34 >= 1      \(3)
c3: x45 + x56 >= 1      \(5)
\
c4: x16 + x23 + x26 + x24 >= 1      \(1,2)
c5: x12 + x26 + x46 + x56 >= 1      \(1,6)
c6: x34 + x12 + x26 + x24 >= 1      \(2,3)
c7: x23 + x24 + x46 + x45 >= 1      \(3,4)
c8: x56 + x46 + x24 + x34 >= 1      \(4,5)
c9: x45 + x46 + x26 + x16 >= 1      \(5,6)
\
c10: x16 + x26 + x24 + x34 >= 1      \(1,2,3)
c11: x23 + x24 + x46 + x56 >= 1      \(1,2,6)
c12: x12 + x26 + x46 + x45 >= 1      \(1,5,6)
```

```
c13: x45 + x46 + x26 + x12 >= 1      \(2,3,4)
c14: x23 + x24 + x46 + x56 >= 1      \(3,4,5)
c15: x34 + x24 + x26 + x16 >= 1      \(4,5,6)
\
\ steiner partition facets, size 3
c16: x12 + x34 + x56 + x26 + x24 + x46 >= 2      \(1,6)(2,3)(4,5)
c17: x23 + x45 + x16 + x26 + x24 + x46 >= 2      \(1,2)(3,4)(5,6)
\
c18: x23 + x34 + x24 + x46 + x56 >= 2      \(1,2,6)(4,5)(3)
c19: x45 + x56 + x23 + x24 + x46 >= 2      \(1,2,6)(3,4)(5)
c20: x45 + x56 + x12 + x26 + x46 >= 2      \(2,3,4)(1,6)(5)
c21: x12 + x16 + x26 + x46 + x45 >= 2      \(2,3,4)(5,6)(1)
c22: x23 + x34 + x24 + x26 + x16 >= 2      \(4,5,6)(1,2)(3)
c23: x12 + x16 + x26 + x24 + x34 >= 2      \(4,5,6)(2,3)(1)
Bounds
0 <= x12 <= 1
0 <= x23 <= 1
0 <= x34 <= 1
0 <= x45 <= 1
0 <= x56 <= 1
0 <= x16 <= 1
0 <= x26 <= 1
0 <= x24 <= 1
0 <= x46 <= 1
End
```

Appendix B

LP of Odd-hole Graph H_5

The LP formulation for the LP relaxation of ILP(5.10) on the odd-hole graph H_5, in an lp form that can be submitted to CPLEX, is the following. Backslashes are comments in CPLEX, and the number in brackets at the end of each line indicates the partition giving rise to that constraint.

Minimize
obj: x12 + x23 + x34 + x45 + x56 + x67 + x78 + x89 + x910 + x110 + x24 + x46 + x68 + x810 + x210
Subject To
\ steiner partition facets, size 2
x12 + x110 >= 1 \(1)
x23 + x34 >= 1 \(2)
x45 + x56 >= 1 \(3)
x67 + x78 >= 1 \(4)
x89 + x910 >= 1 \(5)
\
x110 + x23 + x24 + x210 >= 1 \(1,2)
x12 + x210 + x24 + x34 >= 1 \(2,3)
x23 + x24 + x45 + x46 >= 1 \(3,4)
x34 + x24 + x46 + x56 >= 1 \(4,5)
x45 + x46 + x68 + x67 >= 1 \(5,6)
x56 + x46 + x68 + x78 >= 1 \(6,7)
x67 + x68 + x810 + x89 >= 1 \(7,8)
x78 + x68 + x810 + x910 >= 1 \(8,9)

x89 + x810 + x210 + x110 >= 1 \(9,10)

x12 + x210 + x810 + x910 >= 1 \(1,10)

\

x34 + x24 + x210 + x110 >= 1 \(1,2,3)

x12 + x210 + x45 + x46 >= 1 \(2,3,4)

x23 + x24 + x46 + x56 >= 1 \(3,4,5)

x34 + x24 + x68 + x67 >= 1 \(4,5,6)

x45 + x46 + x68 + x78 >= 1 \(5,6,7)

x56 + x46 + x810 + x89 >= 1 \(6,7,8)

x67 + x68 + x810 + x910 >= 1 \(7,8,9)

x68 + x78 + x210 + x110 >= 1 \(8,9,10)

x89 + x810 + x210 + x12 >= 1 \(9,10,1)

x910 + x810 + x23 + x24 >= 1 \(10,1,2)

\

x110 + x210 + x45 + x46 >= 1 \(1,2,3,4)

x12 + x210 + x46 + x56 >= 1 \(2,3,4,5)

x23 + x24 + x68 + x67 >= 1 \(3,4,5,6)

x34 + x24 + x68 + x78 >= 1 \(4,5,6,7)

x45 + x46 + x89 + x810 >= 1 \(5,6,7,8)

x46 + x56 + x810 + x910 >= 1 \(6,7,8,9)

x67 + x68 + x210 + x110 >= 1 \(7,8,9,10)

x78 + x68 + x12 + x210 >= 1 \(8,9,10,1)

x89 + x810 + x23 + x24 >= 1 \(9,10,1,2)

x810 + x910 + x34 + x24 >= 1 \(10,1,2,3)

\

x110 + x210 + x46 + x56 >= 1 \(1,2,3,4,5)

x12 + x210 + x67 + x68 >= 1 \(2,3,4,5,6)

x23 + x24 + x68 + x78 >= 1 \(3,4,5,6,7)

```
x34 + x24 + x89 + x810 >= 1        \(4,5,6,7,8)
x45 + x46 + x810 + x910 >= 1        \(5,6,7,8,9)
\
\ steiner partition facets, size 3
x12 + x110 + x210 + x24 + x34 >= 2        \(1)(2,3)
x23 + x34 + x24 + x46 + x56 >= 2        \(3)(4,5)
x45 + x56 + x46 + x68 + x78 >= 2        \(5)(6,7)
x67 + x78 + x68 + x810 + x910 >= 2        \(7)(8,9)
x89 + x910 + x12 + x210 + x810 >= 2        \(9)(10,1)
x12 + x110 + x89 + x810 + x210 >= 2        \(1)(9,10)
x23 + x34 + x110 + x24 + x210 >= 2        \(3)(1,2)
x45 + x56 + x23 + x24 + x46 >= 2        \(5)(3,4)
x67 + x78 + x45 + x46 + x68 >= 2        \(7)(5,6)
x89 + x910 + x67 + x68 + x810 >= 2        \(9)(7,8)
\
x12 + x110 + x210 + x45 + x46 >= 2        \(1)(2,3,4)
x23 + x34 + x24 + x68 + x67 >= 2        \(3)(4,5,6)
x45 + x56 + x46 + x810 + x89 >= 2        \(5)(6,7,8)
x67 + x78 + x68 + x210 + x110 >= 2        \(7)(8,9,10)
x89 + x910 + x810 + x23 + x24 >= 2        \(9)(10,1,2)
x12 + x110 + x68 + x78 + x210 >= 2        \(1)(8,9,10)
x23 + x34 + x910 + x810 + x24 >= 2        \(3)(10,1,2)
x45 + x56 + x12 + x210 + x46 >= 2        \(5)(2,3,4)
x67 + x78 + x34 + x24 + x68 >= 2        \(7)(4,5,6)
x89 + x910 + x56 + x46 + x810 >= 2        \(9)(6,7,8)
\
x12 + x110 + x210 + x46 + x56 >= 2        \(1)(2,3,4,5)
x23 + x34 + x24 + x68 + x78 >= 2        \(3)(4,5,6,7)
```

x45 + x56 + x46 + x810 + x910 >= 2 \(5)(6,7,8,9)

x67 + x78 + x68 + x12 + x210 >= 2 \(7)(8,9,10,1)

x89 + x910 + x910 + x34 + x24 >= 2 \(9)(10,1,2,3)

x12 + x110 + x67 + x68 + x210 >= 2 \(1)(7,8,9,10)

x23 + x34 + x89 + x810 + x24 >= 2 \(3)(9,10,1,2)

x45 + x56 + x110 + x210 + x46 >= 2 \(5)(1,2,3,4)

x67 + x78 + x23 + x24 + x68 >= 2 \(7)(3,4,5,6)

x89 + x910 + x45 + x46 + x810 >= 2 \(9)(5,6,7,8)

\

x110 + x23 + x24 + x210 + x45 + x46 >= 2 \(1,2)(3,4)

x12 + x210 + x24 + x34 + x46 + x56 >= 2 \(2,3)(4,5)

x23 + x24 + x45 + x46 + x68 + x67 >= 2 \(3,4)(5,6)

x34 + x24 + x46 + x56 + x68 + x78 >= 2 \(4,5)(6,7)

x45 + x46 + x68 + x67 + x810 + x89 >= 2 \(5,6)(7,8)

x56 + x46 + x68 + x78 + x810 + x910 >= 2 \(6,7)(8,9)

x67 + x68 + x810 + x89 + x210 + x110 >= 2 \(7,8)(9,10)

x78 + x68 + x810 + x910 + x12 + x210 >= 2 \(8,9)(10,1)

x89 + x810 + x210 + x110 + x23 + x24 >= 2 \(9,10)(1,2)

x12 + x210 + x810 + x910 + x24 + x34 >= 2 \(1,10)(2,3)

\

x110 + x23 + x24 + x210 + x46 + x56 >= 2 \(1,2)(3,4,5)

x12 + x210 + x24 + x34 + x68 + x67 >= 2 \(2,3)(4,5,6)

x23 + x24 + x45 + x46 + x68 + x78 >= 2 \(3,4)(5,6,7)

x34 + x24 + x46 + x56 + x810 + x89 >= 2 \(4,5)(6,7,8)

x45 + x46 + x68 + x67 + x810 + x910 >= 2 \(5,6)(7,8,9)

x56 + x46 + x68 + x78 + x210 + x110 >= 2 \(6,7)(8,9,10)

x67 + x68 + x810 + x89 + x210 + x12 >= 2 \(7,8)(9,10,1)

x78 + x68 + x810 + x910 + x23 + x24 >= 2 \(8,9)(10,1,2) x89 + x810 + x210 + x110 + x34 + x24 >= 2 \(9,10)(1,2,3)

x12 + x210 + x810 + x910 + x45 + x46 >= 2 \(1,10)(2,3,4)

\

x110 + x23 + x24 + x210 + x68 + x67 >= 2 \(1,2)(3,4,5,6)

x12 + x210 + x24 + x34 + x68 + x78 >= 2 \(2,3)(4,5,6,7)

x23 + x24 + x45 + x46 + x89 + x810 >= 2 \(3,4)(5,6,7,8)

x34 + x24 + x46 + x56 + x810 + x910 >= 2 \(4,5)(6,7,8,9)

x45 + x46 + x68 + x67 + x210 + x110 >= 2 \(5,6)(7,8,9,10)

x56 + x46 + x68 + x78 + x12 + x210 >= 2 \(6,7)(8,9,10,1)

x67 + x68 + x810 + x89 + x23 + x24 >= 2 \(7,8)(9,10,1,2)

x78 + x68 + x810 + x910 + x34 + x24 >= 2 \(8,9)(10,1,2,3)

x89 + x810 + x210 + x110 + x45 + x46 >= 2 \(9,10)(1,2,3,4)

x12 + x210 + x810 + x910 + x46 + x56 >= 2 \(1,10)(2,3,4,5)

\

x34 + x24 + x210 + x110 + x68 + x67 >= 2 \(1,2,3)(4,5,6)

x12 + x210 + x45 + x46 + x68 + x78 >= 2 \(2,3,4)(5,6,7)

x23 + x24 + x46 + x56 + x810 + x89 >= 2 \(3,4,5)(6,7,8)

x34 + x24 + x68 + x67 + x810 + x910 >= 2 \(4,5,6)(7,8,9)

x45 + x46 + x68 + x78 + x210 + x110 >= 2 \(5,6,7)(8,9,10)

x56 + x46 + x810 + x89 + x210 + x12 >= 2 \(6,7,8)(9,10,1)

x67 + x68 + x810 + x910 + x23 + x24 >= 2 \(7,8,9)(10,1,2)

x68 + x78 + x210 + x110 + x34 + x24 >= 2 \(8,9,10)(1,2,3)

x89 + x810 + x210 + x12 + x45 + x46 >= 2 \(9,10,1)(2,3,4)

x910 + x810 + x23 + x24 + x46 + x56 >= 2 \(10,1,2)(3,4,5)

Bounds

0 <= x12 <= 1

0 <= x23 <= 1

0 <= x34 <= 1

0 <= x45 <= 1

0 <= x56 <= 1

0 <= x67 <= 1

0 <= x78 <= 1

0 <= x89 <= 1

0 <= x910 <= 1

0 <= x110 <= 1

0 <= x24 <= 1

0 <= x46 <= 1

0 <= x68 <= 1

0 <= x810 <= 1

0 <= x210 <= 1

End

Chapter 6

Conclusion

In this dissertation, we examined some specific network connectivity problems, with the goal of achieving improved approximation algorithm and integrality gap results for these problems. We also introduced an important new, highly useful and applicable, network connectivity problem, and presented an approximation algorithm and integrality gap for it.

In Chapter 2, we introduced the new vital core connectivity problem (VCC). We showed that it is NP-hard, but that there exists a polynomial-time algorithm that solves EMST (the special case of VCC in which $k = l = 1$). In addition, we provided a complete linear description for EMST. Chapter 2 is the first paper to introduce EMST, as well as the first to solve it. In Chapter 2, we also presented the first constant factor approximation algorithm for VCC, as well as the first upper bound on the integrality gap of the LP relaxation of VCC. In particular, we showed an approximation guarantee (and upper bound on the integrality gap) of $\frac{8}{3}$ for $l \geq \lceil \frac{k}{2} \rceil$; of $\frac{19}{6}$ for $2 \leq l < \lceil \frac{k}{2} \rceil$; and of $\frac{5}{2}$ 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. We took note of our approximation guarantee (and upper bound on the integrality gap) for several special cases. For example, for the special case VCC(k, k, V^*), $V^* \neq V$, we have 2; for the special case VCC$(2, 1, V^*)$ (in which the secondary vertices are not a cut set), we have $\frac{3}{2}$. The special case of VCC$(2, 1, V \backslash \{r\})$ finds a feasible multi-2EC solution in which r is not a cut vertex, and for it our results give a $\frac{3}{2}$ approximation guarantee (and upper bound on the integrality gap). This extended our $\frac{3}{2}$-approximation algorithm for multi-2EC in Chapter 4. Finally, we provided the first complete linear description of, and polynomial-time algorithm for, an extension of EMST to numerous disjoint vital cores.

In Chapter 3, we examined the NP-hard problem multi-SNDP, and presented results which are improvements over the current best known results for certain cases of it. In order to achieve this, we presented some results on minimum cost perfect matchings, T-joins, and minimum cost spanning trees. Using these results, we showed that the integrality gap for the LP relaxation of multi-SNDP is at most $\frac{3}{2}\frac{r_m}{r_l}$ when r_m is even, and at most $\frac{3}{2}\frac{r_m}{r_l} + \frac{1}{2r_l}$ when r_m is odd, where r_m is the maximum vertex connectivity requirement and r_l is the smallest nonzero vertex connectivity requirement. Along with this, we presented an approximation algorithm for multi-SNDP which has an approximation guarantee the same as these upper bounds. These are improvements for the best known bounds and approximation guarantees for certain special cases of multi-SNDP, and we presented a family of examples demonstrating this.

In Chapter 4, we examined multi-SNDP, SNDP, VC-SNDP, and other variations, with low vertex connectivity requirements. We presented extensions and applications of our results in Chapter 3 to related problems, and we investigated cases where the optimal values of multi-SNDP and SNDP are equal. Furthermore, we demonstrated that multi-SNDP and metric SNDP are closely related for $r_i \in \{0, 1, 2\}$, $i \in V$; and we showed that the upper bound for the integrality gap of the LP relaxation of metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, is $3/2$, thus extending the known upper bound of $3/2$ for the integrality gap of the LP relaxation of metric 2EC [1]. We also showed that our multi-SNDP approximation algorithm that was presented in Chapter 3, with certain modifications, applies as well for metric SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$, thus giving a $\frac{3}{2}$-approximation algorithm for that case. We then moved on and investigated the related VC-SNDP, particularly for small vertex connectivity requirements. We further modified our approximation algorithm to obtain a $\frac{3}{2}$-approximation algorithm for metric VC-SNDP with $r_i \in \{0, 1, 2\}$, $i \in V$. We also showed that there is a $\frac{3}{2}$ upper bound for the integrality gap of the LP relaxation of this problem. Our $\frac{3}{2}$ approximation guarantee and integrality gap bound applies to the Steiner tree problem with either uniform or metric edge costs.

In the last chapter, Chapter 5, we investigated different ILP formulations for the Steiner tree problem (ST), and their corresponding LP relaxations. We considered finding stronger LP relaxations for ST in terms of the integrality gap of the LP relaxation. In particular, we considered finding LP relaxations for ST such that the LP relaxation, when restricted to MST, has an integrality gap of 1. We proposed two new formulations for ST, one based on the undirected cut formulation and the other on Steiner partitions, and investigated their strength in terms of their associated integrality gaps.

6.1 Future Work

There are a number of open problems and avenues for future research.

It remains an open problem to find an ILP for the Steiner tree problem such that its LP relaxation has an integrality gap of 1.39 (or, at the very least, better than the current integrality gap of 1.55 [3]), thus matching the current best known approximation guarantee of 1.39 for ST [3]. It also remains an open problem to determine whether or not the integrality gap of the bi-directed cut relaxation for ST is less than 2 [3].

Further work on the k vertex out-connected from r spanning subgraph problem

(kVC$\{r\}$), briefly considered at the end of Chapter 4, could be done. This problem is of particular interest, since the 2-approximation algorithm for 3VC in [2] calls a subroutine that returns a feasible solution of kVC$\{r\}$ with total cost at most twice the optimal value [4]. If the approximation guarantee for kVC$\{r\}$ can be improved, then the best known approximation guarantee of 2 for 3VC can also be improved.

The 2-approximation algorithm (and upper bound on integrality gap) for multi-SNDP and SNDP comes from Jain's results using iterated LP rounding [6]. Although it runs in polynomial time, Jain's algorithm makes use of the Ellipsoid method, which in practice is prohibitive. It remains open to find for SNDP an approximation algorithm that does not use the Ellipsoid method and that has an approximation guarantee of 2 or better. It is also a long-standing open problem to obtain any constant factor approximation algorithm for VC-SNDP. The best result so far connected to this is a 2-approximation algorithm for VC-SNDP with vertex connectivity requirements $r_i \in \{0, 1, 2\}$, $i \in V$ [5], and a k-approximation algorithm for kVC [8]. It has been shown that iterative rounding will not work for VC-SNDP [5]. An intermediate problem is the element connectivity problem [7], in which only edges and certain vertices can fail. Problem SNDP is a special case of this problem. The best known approximation guarantee for the element connectivity problem is 2, and is obtained from iterative rounding [5].

Problem VCC that we introduced in this dissertation can be generalized as follows: Let $G = (V, E)$ be a multi-graph with non-negative edge costs $c \in \mathbb{R}^E$ and vertex connectivity requirements $r_i \in \mathbb{N}_{\geq 0}$ for all $i \in V$, and $t_i \in \mathbb{N}_{\geq 0}$ for all $i \in V^*$ (so vertices in V^* each have two vertex connectivity requirements). Let $r_{ij} = \min\{r_i, r_j\}$ for each pair of distinct vertices $i, j \in V$, and let $t_{ij} = \min\{t_i, t_j\}$ for each pair of distinct vertices $i, j \in V^*$. The problem is to find a minimum-cost subgraph that has, for each pair of distinct vertices $i, j \in V$, at least r_{ij} edge-disjoint paths in G between i and j; and that also has, for each pair of distinct vertices $i, j \in V^*$, at least t_{ij} edge-disjoint paths in $G[V^*]$ between i and j. We call this the *survivable network* *vital core connectivity problem* (SNVCC). It is easy to see that SNVCC is NP-hard. Problem SNVCC can be generalized even further to a problem which includes Jain's general problem as a special case. We call this the *vital core network connectivity problem* (CORE), and define it as follows: Let $G = (V, E)$ be a multi-graph with (1) non-negative edge costs $c \in \mathbb{R}^E$, (2) vertices partitioned into vital vertices $V^* \subseteq V$ (which form a "vital core") and secondary vertices $V \backslash V^*$, (3) given weakly supermodular cut requirement functions $f : 2^V \longrightarrow \mathbb{Z}$ and $g : 2^{V^*} \longrightarrow \mathbb{Z}$, and (4) a given edge capacity function $u : E \longrightarrow \mathbb{Z}_{\geq 1}$, where $u_e := u(e)$ is an upper bound on the number of copies of edge $e \in E$ that may be used by a solution. The problem is to find a minimum cost spanning multi-subgraph of G such that there are at least $f(S)$ edges in every cut $\delta_G(S)$, for all $S \subseteq V$, and at least $g(S')$ edges in every cut $\delta_{G[V^*]}(S')$, for all $S' \subseteq V^*$. Although this problem arises naturally in many applications, and has some similarities with other problems, as well as encompasses a great number of problems, this problem has not yet been studied. If it can be shown that every basic solution x^* of CORE's corresponding LP relaxation contains an edge $e' \in E$ such that $x^*_{e'} \geq 1/2$, then it is straightforward to show that Jain's iterated LP rounding method [6] can be extended to CORE to obtain a 2-approximation algorithm for CORE (and thus for SNVCC as well, since SNVCC is a special case of CORE). More work needs to be done on this to determine if such an edge always exists in a basic solution of the LP relaxation of CORE. Work could be done as well investigating whether other approaches yield any approximation algorithm for CORE.

Bibliography

[1] A. Alexander, S. Boyd, P. Elliott-Magwood, On the Integrality Gap of the 2-Edge Connected Subgraph Problem, Technical Report TR-2006-04, S.I.T.E., University of Ottawa, Ottawa (2006).

[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] J. Byrka, F. Grandoni, T. Rothvoβ, L. Sanità, An Improved LP-based Approximation for Steiner Tree, STOC 2010, Cambridge, Massachusetts, USA (2010).

[4] Y. Dinitz, Z. Nutov, A 3-Approximation Algorithm for Finding Optimum 4,5-Vertex-Connected Spanning Subgraphs, Journal of Algorithms 32 (1999) 31-40.

[5] L. Fleischer, K. Jain, D. Williamson, Iterative Rounding 2-approximation Algorithms for Minimum-cost Vertex Connectivity Problems, Journal of Computer and Systems Sciences 72 (2006) 838-867.

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

[7] K. Jain, I. Mandoiu, V. V. Vazirani, D.P. Williamson: A Primal-dual Schema Based Approximation Algorithm for the Element Connectivity Problem, Pro-

ceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms (1999) 484-489.

[8] G. Kortsarz, Z. Nutov, Approximating Node Connectivity Problems via Set Covers, Algorithmica 37 (2003) 75-92.