HUB & REGENERATOR LOCATION AND SURVIVABLE NETWORK DESIGN

A DISSERTATION SUBMITTED TO

THE DEPARTMENT OF INDUSTRIAL ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

By

Onur Özkök

December, 2010

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Assoc. Prof. Dr. Oya Ekin Karaşan (Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Assoc. Prof. Dr. Hande Yaman Paternotte (Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Prof. Dr. Selim Aktürk

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Prof. Dr. İmdat Kara

ABSTRACT

HUB & REGENERATOR LOCATION AND SURVIVABLE NETWORK DESIGN

Onur Özkök
Ph.D. in Industrial Engineering
Supervisor: Assoc. Prof. Dr. Oya Ekin Karaşan
Supervisor: Assoc. Prof. Dr. Hande Yaman Paternotte
December, 2010

With the vast development of the Internet, telecommunication networks are employed in numerous different outlets. In addition to voice transmission, which is a traditional utilization, telecommunication networks are now used for transmission of different types of data. As the amount of data transmitted through the network increases, issues such as the survivability and the capacity of the network become more imperative. In this dissertation, we deal with both design and routing problems in telecommunications networks. Our first problem is a two level survivable network design problem. The topmost layer of this network consists of a backbone component where the access equipments that enable the communication of the local access networks are interconnected. The second layer connects the users on the local access network to the access equipments, and consequently to the backbone network. To achieve a survivable network, one that stays operational even under minor breakdowns, the backbone network is assumed to be 2-edge connected while local access networks are to have the star connectivity. Within the literature, such a network is referred to as a 2-edge connected/star network. Since the survivability requirements of networks may change based on the purposes they are utilized for, a variation of this problem in which local access networks are also required to be survivable is also analyzed. The survivability of the local access networks is ensured by providing two connections for every component of the local access networks to the backbone network. This architecture is known as dual homing in the literature. In this dissertation, the polyhedral analysis of the two versions of the two level survivable network design problem is presented; separation problems are analyzed; and branch-and-cut algorithms are developed to find exact solutions.

The increased traffic on the telecommunications networks requires the use of high capacity components. Optical networks, composed of fiber optical cables, offer solutions with their higher bandwidths and higher transmission speeds. This makes the optical networks a good alternative to handle the rapid increase in the data traffic. However, due to signal degradation which makes signal regeneration necessary introduces the regenerator placement problem as signal regeneration is a costly process in optical networks. In the regenerator placement problem, we study a location and routing problem together on the backbone component of a given telecommunications network. Survivability is also considered in this problem simultaneously. Exact solution methodologies are developed for this problem: mathematical models and some valid inequalities are proposed; separation problems for the valid inequalities are analyzed and a branch-and-cut algorithm is devised.

Keywords: Survivable Network Design, 2-edge Connectedness, Dual Homing, Regenerator Location, Polyhedral Analysis, Branch-and-Cut Algorithm.

ÖZET

ERİŞİM CİHAZI - GÜÇLENDİRİCİ YERSEÇİMİ VE KALIMLI AĞ TASARIMI

Onur Özkök
Endüstri Mühendisliği, Doktora
Tez Yöneticisi: Assoc. Prof. Dr. Oya Ekin Karaşan
Tez Yöneticisi: Assoc. Prof. Dr. Hande Yaman Paternotte
Aralık, 2010

İnternet kullanımının yaygınlaşması ile birlikte haberleşme ağları, geleneksel kullanım alanı olan ses iletiminin yanısıra her türlü veri iletimi için kullanılmaya başlanmıştır. İletilen veri miktarlarının yüksek olması, hem ağların kalımlılığının, hem de ağ üzerindeki bileşenlerin iletilen veri miktarlarına uygun kapasitelere sahip olmalarının önemini artırmıştır. Bu tezde hem haberleşme ağı tasarımı hem de haberleşme ağları üzerinde rotalama problemleri üzerinde durmaktayız. İncelediğimiz ilk problem bir katmanında erişim cihazlarını birbirine bağlayan omurga ağının, diğer katmanında ise kullanıcıları erişim cihazlarına bağlayan yerel erişim ağlarının bulunduğu kalımlı ağ tasarımı problemidir. Kalımlı ağ, meydana gelebilecek bir arıza durumunda ağın veri iletimine devam edebilmesi olarak tanımlanabilir. Oluşturulacak ağın kalımlı olması için omurga ağı 2-ayrıt bağlı, yerel erişim ağları ise yıldız mimarisinde tasarlanacaktır. Bu ağ yapısı literatürde 2-ayrıt bağlı/yıldız ağ olarak adlandırılmaktadır. Haberleşme ağlarının kullanıldıkları yere göre güvenilirlik gereksinimleri değişebildiği için bu problemin yerel erişim ağlarının da kalımlı olduğu bir çeşitlemesi de tez kapsamında incelenecektir. Yerel erişim ağlarının kalımlı olması, literatürde ikili atama olarak bilinen, her kullanıcının iki erişim cihazına bağlandığı bir mimari kullanılarak sağlanacaktır. Tez kapsamında iki kalımlı ağ tasarımı probleminin çeşitlemeleri için çokyüzlü analizi gerçekleştirilecek, ilgili ayırma problemleri incelenecek ve eniyi çözümlerinin bulunması için dal-kesi algoritmaları geliştirilecektir.

Haberleşme ağları üzerindeki veri trafiğinin artması ağların kurulumunda yüksek kapasiteye sahip bileşenlerin kullanımını gerekli kılmaktadır. Fiber optik kablolardan oluşan optik ağlar, sahip oldukları yüksek bant genişlikleri ve yüksek veri iletim hızlarına imkan tanımaları sayesinde, artan veri trafiği konusunda bir çözüm alternatifi olmaktadır. Ancak sinyallerin yeniden üretilmesini gerektiren sinyal zayıflama problemi nedeniyle güçlendirici yerseçimi problemi ortaya çıkmaktadır. Bu problem özellikle sinyallerin yeniden yaratılmasının yüksek maliyeti nedeniyle optik ağlarda önem kazanmaktadır. Güçlendirici yerseçimi probleminde mevcut bir haberleşme ağının omurga bileşeni üzerinde rotalama ve yerseçimi problemleri birlikte çözülmektedir. Bu problemde aynı zamanda ağın kalımlı olması gerekliliği de dikkate alınmaktadır. Bu problemin eniyi çözümünün bulunması amacıyla, matematiksel modeller kurulmuş, bazı geçerli eşitsizlikler önerilmiş, bu eşitsizliklere ait ayırma problemleri incelenmiş ve bir dal-kesi algoritması geliştirilmiştir.

Anahtar sözcükler: Kalımlı Ağ Tasarımı, 2-ayrıt Bağlılık, İkili Atama, Güçlendirici Yerseçimi, Çokyüzlü Analizi, Dal-kesi Algoritması.

Acknowledgement

I would like to express my sincere gratitude to my supervisors, Assoc. Prof. Oya Ekin Karaşan and Assoc. Prof. Hande Yaman Paternotte, for which this doctoral study would not have started, continued, and completed without their endless support and encouragement. I am grateful to them for believing in me more than I did.

I am also profoundly grateful to Prof. Ali Ridha Mahjoub, to whom I and this work owe much. This dissertation and myself indeed profitted a lot from his ideas and experience, as well as his wishes of "Bon Courage". I feel privileged to have the opportunity to work with him.

My sincere thanks also goes to Prof. İmdat Kara, who has always supported and guided me since my interview at Başkent University. He was always there at all happy and sad events and I felt his support at all times.

I am indebted to the members of my dissertation committee, Prof. Selim Aktürk, Prof. Barbaros Tansel, and Assoc. Prof. Mustafa Akgül, for accepting to be a member of my committee and to read and review this thesis. Their comments and recommendations have been very helpful.

I would like to thank our department chair Prof. İhsan Sabuncuoğlu and Prof. Ülkü Gürler, chair of the graduate committee, for giving me their support when I decided to be a full time member of our department.

I also would like to thank TÜBİTAK, who supported project 107M247 and my research during my Ph.D. study.

My colleagues at Başkent University deserve many thanks. Specifically, I would like to thank Dr. Tolga Bektaş, Dr. Yazgı Tütüncü, Dr. Ergün Eraslan, Buğra Çamlıca, and Demiralp Hatipoğlu for their support and friendships. Additional thanks are due to all my friends at Bilkent University, and in specific to Dr. Sibel Alumur and Yahya Saleh.

I am also indebted to Asst. Prof. Alper Şen, who helped me to realize my dream about my professional career.

I am proud of being a member of Bilkent and Başkent Universities, and would like to thank each and every member of both.

I would like to take this opportunity to express my thanks to Prashant Soral of Applied Materials, for he has been so kind and understanding to me since our first interview and has provided me with a great opportunity to start my professional career.

I am mostly indebted to my parents Veli and Meliha, my brother Halil and my sister Esra for all their endless support, care and love. Without them I would not be the person I am. I would also like to express my thanks to my parents-in-law Avni and Birsen Altınova and my brother-in-law Hakan.

I am very grateful to my beloved wife and everlasting love, Sezen, who has always been understanding, motivating and full of love. Her support meant really much.

Finally, I would like to express my love and thanks to my little son, Tunçalp, whose arrival has started a series of very happy events.

To my wife . . .

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The structure and use of communication networks have significantly changed in the last years. While communication networks were mainly dedicated to telephones in the past, currently voice, video and other types of data are transmitted through communication networks. This resulted in a huge increase in the traffic flow over the communication networks. It has been estimated that, the traffic doubles every ten months, and is expected that the increase will continue [18]. Moreover, the users of communication networks are getting more and more demanding in the service quality they receive. The term, *Quality of Service* (QoS), which depends on a few factors, such as availability and losses, is used to evaluate the service provided to the users. QoS could be defined as maintaining the performance of the service provided at the requested level.

The building blocks of the network such as links and nodes are subject to failure. If the amount of data flowing through them is large, a failure may result in significant losses. However, users do not want to experience any loss, so a network which has QoS should not allow data losses due to failures of network components. Obviously, the QoS depends greatly on the design of the network and this makes the network design problem very important.

Survivability of a network can be defined as the ability of transmitting the data even in case of a failure. This is usually achieved by having sufficiently many disjoint paths, which are paths without common links and/or nodes between every pair of nodes that are to communicate. A network is referred to as *k-edge connected* if there are at least k-edge disjoint paths between every node pair. In other words, k is the minimum number of edges the removal of which disconnects the network. If a failure makes a communication path unusable, other communication paths could still be used. The degree of survivability or the number of disjoint paths required depends on the properties of the application for which the network is designed. For example, while a fully connected structure is preferred for some networks, a two-edge connected network might be sufficient for others. As it can be seen easily, the cost of the network increases as the level of survivability increases, since the number of necessary links to be used in the network increases. Therefore there is a trade-off between the cost of constructing the network and the survivability of the network. This trade-off should be considered while determining the required level of survivability.

Although there are many layers in modern communication networks, two layers are of great interest for the design process. The first is the *access network* which connects the users to the hubs. The hubs are the centers which provide the service required by the users. In this context, the hubs may be concentrators, switchers, multiplexers, etc. Hereafter, concentrator is also going to be used to refer to hubs. Similarly, we will use the term terminal to refer to the users. The hubs are either inter-connected or connected to a central unit. The network that connects the hubs is called the *backbone network*. Since the data transmission between nodes is performed through the hubs, the backbone network is crucial for the communication networks. Structure of a communication network can be specified by the architectures of its backbone and access networks [29]. For example, a fully connected/tree network is composed of a fully connected backbone network and tree access network. Some examples are provided in Figures 1.1-1.3.

In all figures, the hubs and users are represented by squares and circles, respectively. The backbone edges are shown by lines while dashed lines are used for assignments of users to hubs.

In this study, we are interested in location and survivable network design

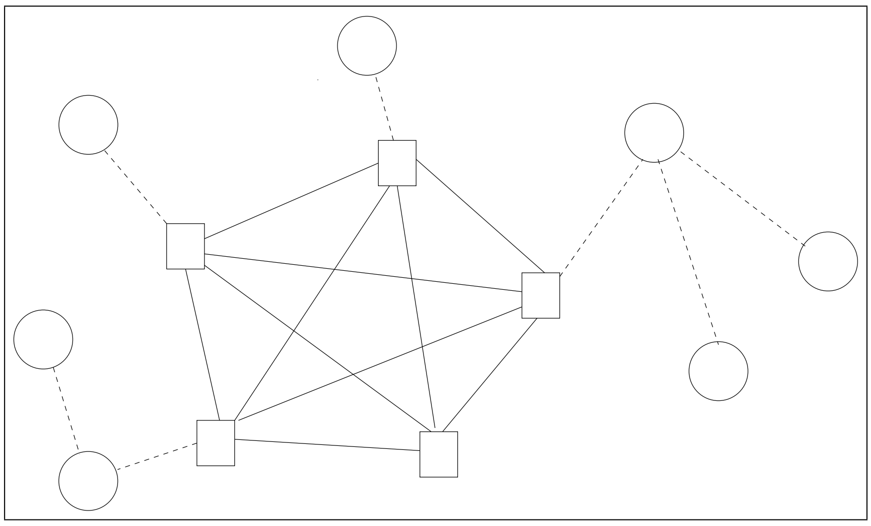

Figure 1.1: A fully connected/tree network

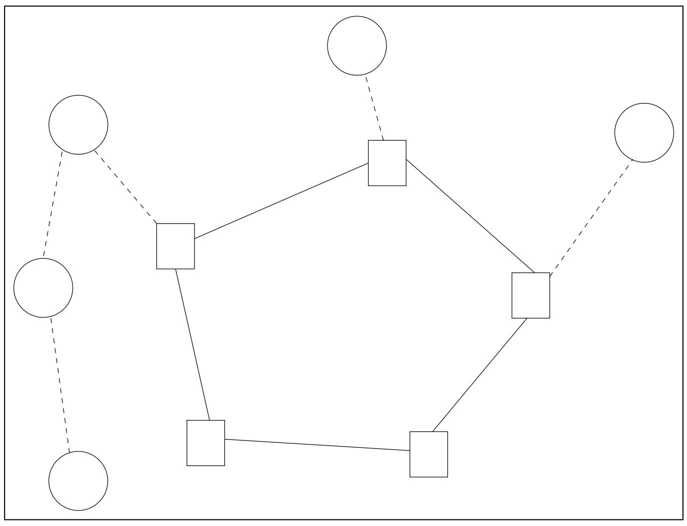

Figure 1.2: A ring/path network

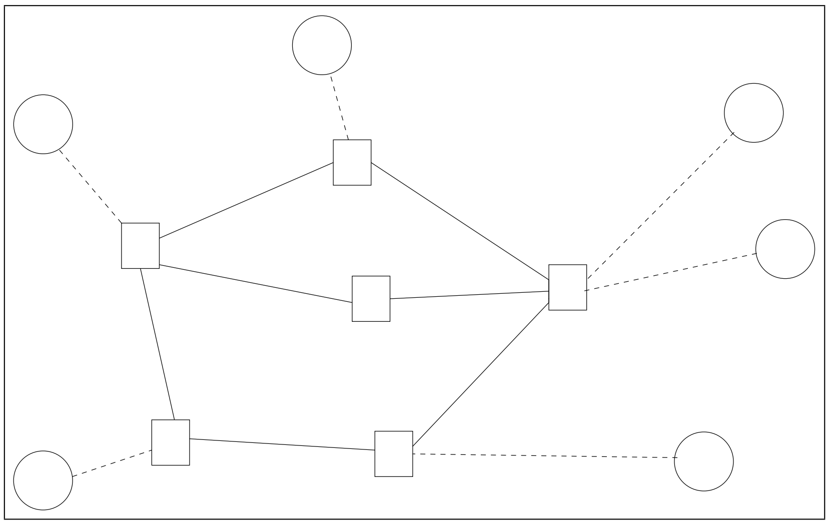
Figure 1.3: A 2-edge connected/star network

problems. We assume that failures occur only on links and at most one link can fail at a time. To attain sufficient level of survivability we consider a network that consists of a 2-edge connected backbone network and star access networks, in which all terminals are directly connected to hubs. Based on the structures of the backbone and access networks, this problem is called *2-edge connected/star network design problem* or *2-edge connected/star subgraph problem* (2ECSSP in short). The hubs of the backbone network will be chosen among the users and the remaining users which are not selected as hubs will be assigned to hubs. It is assumed that if a user is selected as a hub, it is assigned to itself. There is a cost for establishing a link between two hubs, and similarly there is an assignment cost when a user is assigned to a hub. We assume that there is a root node, which is chosen to be a hub in the graph. This root node may be a special unit which connects the backbone network to other communication networks. However, the assumption of the existence of a root node may be relaxed. In this problem all links and hubs are assumed to have infinite capacities and the objective is to construct such a network with minimum cost.

Note that in 2ECSSP, we only focus on the survivability in the backbone network. Naturally, the survivability of the backbone is very important as the data transmission is performed through this part. However, it may be noticed that in case of a failure in the link that connects a terminal to a concentrator, the terminal will not have a connection to the backbone, i.e. it will be disconnected from the network. This may not constitute a problem as critical as the failures in the backbone network. However, as a user is only interested in the service he/she receives, the survivability of the backbone will not mean anything if his/her connection is lost. Therefore, considering the survivability of both the backbone and the access networks at the same time may be necessary. For this reason, we also consider a variant of 2ECSSP in which the backbone is again 2-edge connected while each terminal, on which no concentrator is installed, is assigned to two concentrators instead of one. Although this modification breaks the star architecture of the access networks, the problem is still similar to 2ECSSP. We will refer to this problem as the *2-edge connected subgraph with dual homing problem* (2ECSDHP in short). In both variants of the two level survivable network design problems the following tasks should be completed to construct a solution:

i) partition of nodes as hubs and non-hubs,

ii) design of the backbone network,

iii) design of the access networks.

The design of the access networks is trivial if locations of the hubs are known. However, since it affects the design of the backbone network all three problems must be solved simultaneously.

The increased traffic on the telecommunications networks requires the use of high capacity components. Optical networks, composed of fiber optical cables, offer solutions with their higher bandwidths and higher transmission speeds. This makes the optical networks a good alternative to handle the rapid increase in the data traffic. However, signal degradation makes signal regeneration necessary in optical networks and this introduces the regenerator placement problem since signal regeneration is a costly process.

Survivability is not the only factor for the Quality of Service (QoS) of a telecommunications network. Since the traffic on the telecommunications networks is rapidly increasing due to the new applications and increased number of users, telecommunications networks have to have sufficient capacity. This highlights the capacity of the components of the network as another crucial aspect for the QoS. Optical networks consisting of fiber optic cables offer higher bandwidths and higher transmission speeds, and hence they could handle the increasing traffic on the Internet (see [47]). This makes the fiber optic cables an important component alternative in the construction of communications networks. The explosive growth in Internet traffic fed by the increase in the number of users and resources consumed by modern applications necessitates the use of such high capacity optical communication networks. However, a transmission impairment, namely signal degradation, makes some signal routes unusable [41]. Optical signals are degraded during emission from a node, in other words, the *signal-to-noise* ratio (SNR) of the signal which may be considered as the quality of the signal reduces after it is emitted. If SNR is below a threshold value, the signal cannot be used, so an acceptable SNR level, depending on the properties of the optical network, needs to be provided at the receiver node (see [41]). Therefore, signal degradation appears as another issue that should be considered in the telecommunications network design.

The amount of the degradation depends on the length of the fiber optic cable. To overcome this, signal regenerators which can increase SNR, are used to increase the transmission range. However, as such devices and the regeneration process have high costs, it is desired to minimize the number of regenerators used in a network. This problem will be referred to as *Regenerator Placement Problem* (RPP in short) throughout the text. Although signal degradation also occurs in electrical signals, we focus on the optical networks. This is because the regeneration of optical signals may be much more complicated than the regeneration of electrical signals. Why there is such a difference will be explained in the next chapter.

Unlike the first two problems, in the regenerator placement problem we are not directly interested in a network design problem. On the contrary, we assume that there is an already designed network on which the RPP will be solved. In the 2ECSSP and the 2ECSDHP, we do not consider the routing of the signals. This is because the routing problem, which is equivalent to the problem of finding 2-edge connected paths between two nodes, becomes simple once the network is designed. The reason for this is any path will be feasible for data transmission. However, this is not the case in the RPP due to signal regeneration requirement. Note that a path is feasible if there are regenerators on the appropriate nodes of the path which will allow the signal to be regenerated before its SNR gets too low. Therefore, the locations of the regenerators must be taken into account in the routing of the signals. This makes the routing problem not a trivial one, since we need to consider some side constraints such as length of the path and location of the regenerators. These side constraints will be explained in detail in Chapter 6 where we define the RPP formally.

Therefore the RPP consists of the following components:

- Determining the number and location of regenerators.
- Routing of the signals.

We only consider the backbone network in this problem and assume a 2-edge connected backbone as the underlying graph. For the survivability we stick to the 2-edge connected structure. It should be noted that the side constraints described before also affect the survivability of the network since they may make some paths unusable. Although the underlying network is known to be 2-edge connected, it does not guarantee that there are at least 2-edge disjoint feasible paths between every node pair due to the side constraints. As survivability is still a necessary property, we take the survivability requirement into account while solving the location and routing problems. Again we need to solve these subproblems simultaneously to achieve the optimal solution.

The rest of this thesis is organized as follows. In the next chapter we will review the literature related to the three problems described above. In Chapter 3, we study 2ECSSP and provide the polyhedral analysis we have done. The solution methodology we proposed and the computational results for the 2ECSSP are presented in Chapter 4. The polyhedral studies and computational results about the 2ECSDHP can be found in Chapter 5. We present the analysis of the RPP in Chapter 6. Finally, in Chapter 7, we conclude the thesis with the conclusive remarks and discussion on the future research directions.

Chapter 2

Literature Survey

The first two problems we are interested in are closely related to two problems, namely survivable network design problem and hub location problem, since we both design a survivable backbone network and we also solve a hub location problem to choose hub nodes. Note that if hub nodes are known a priori, then the problem decomposes into sub-problems, which are the problems of designing the backbone and access networks. We should note here that the latter is less important than the first, since the designing of the access networks is trivial once the location of hubs are determined as the architecture chosen for the access networks imposes that each user is assigned to the nearest hub. Therefore, our problem is actually a two level network design problem. In the first level the hubs are determined while the networks are designed in the second level. Both problems have been widely studied in the literature individually. Therefore, it would be beneficial to review the literature on these subjects.

Gourdin et al. [18] reviewed the studies in telecommunication context which include location problems. Since there are many devices which could be considered as hubs in telecommunication systems, location problems arise very frequently in this area. As the definition of each problem depends significantly on the properties of the network and its components, there are many variations of hub location problems in telecommunication literature. The problems could be divided into two main classes, uncapacitated (UCLP) and capacitated concentrator location (CCLP) problems. The cost components and requirements which must be satisfied by the network also vary in hub location problems. For example, while routing cost of the traffic on the network is considered in some problems, such costs might be ignored in others. In the review of Gourdin et al. [18], many variations of hub location problems are discussed. The mathematical models and the methods utilized to solve the problems are described in the article. The authors also include the studies on the polyhedral structures of the models and some heuristics used to solve them.

Yaman [45] is interested in the design of a fully connected/star network. The traffic between node pairs is considered so the arc capacities become important for the solution. In this study a mathematical formulation which determines the hub nodes from a node set and the capacities of the arcs is provided. The aim is to minimize the total cost of installing capacities to arcs and setup cost of hubs. As it can be seen from the objective, the cost of installing edges of the backbone network is ignored. So the problem is closer to hub location problems rather than survivable network design problems. Polyhedral structure of this problem is analyzed and some valid and facet defining inequalities are proposed.

Another hub location problem is analyzed by Labbe et al. [32]. In their article, a fully connected/star network is designed and the hubs are chosen among the given set of nodes. The traffic between nodes is considered and sufficient capacities are installed to the arcs. Although the routing and hub installing costs are considered, the cost of edges in the backbone networks is ignored. Since the hubs are not known a priori, the objective function is quadratic and the problem is referred to as *Quadratic Capacitated Hub Location Problem with Single Assignment* (QHL). Different capacity structures are analyzed and some variants of QHL are obtained. Several formulations for the problems are also proposed. Polyhedral structures of the problems are studied and a branch and cut algorithm is developed based on the polyhedral results. A comparison between the formulations is also provided.

The 2-edge connected subgraph problem is a special case of the survivable network design problem. A survey on survivable network design is presented by Kerivin and Mahjoub [25]. Since the 2-edge connected subgraph problem is NP-hard [34], survivable network design problem which includes the first one is also NP-hard. However, there are some special cases of the survivable network design problem which can be solved in polynomial time. The authors discuss such cases and also review heuristics and approximation algorithms. An integer linear program is proposed and the associated polytope is analyzed. Valid inequalities for the problems and their separation algorithms are also described. The polytopes for edge/node survivable network problems are discussed for special graphs. The concept of critical extreme points which can be used to improve linear programming relaxations is also reviewed. In this study, some branch-and-cut algorithms from the literature are devised for these problems and the numerical results obtained via these methods are also discussed. If the lengths of the disjoint paths of the subgraph are bounded from above, a variation of the network design problem, namely, length constrained survivable network design problem, arises. This problem is also reviewed in the survey. Lastly the authors classify design problems arising under capacity restrictions which are not widely studied in the literature as further research areas.

The survivable network design problem has many other variations. Stoer [40] describes some of these variations and provide integer programming formulations. Classes of valid inequalities are proposed and lifting procedures are described by the author. The inequalities are analyzed to identify those which are facet defining. The cutting plane algorithms and separation procedures are also included. Grötschel et al. [21] analyze the survivable network design problem, too. Different types of survivable network design problems and mathematical formulations are presented. The authors discuss the polyhedral aspects of the problems and provide cutting plane algorithms for solving them. Heuristic algorithms are also described as in general most survivable network design problems are NP-hard necessitating the use of efficient heuristics. Computational results on data from the literature are also included.

As stated before, 2-edge connected network design is a special case of the survivable network design problem. In this problem, the aim is to find a spanning subgraph, in which there are at least 2-edge disjoints paths between every node pair, such that the sum of edge weights is minimum. It can be easily seen that if the locations of the hubs are known, the access network design problem is trivial and hence 2ECSSP reduces to 2-edge connected subgraph problem. For this reason, 2-edge connected subgraph problem is of great interest for the 2ECSSP. The polytope associated with this problem, namely 2-edge connected subgraph polytope is analyzed and described by Mahjoub [34]. This paper discusses the facets of the associated polytope and necessary conditions for them to be facet defining. There are two main classes of inequalities of this polytope. The first class is the *trivial inequalities*, which refers to the lower and upper bounds of the variables. The second class is the *cut inequalities* used to make sure that the subgraph is 2-edge connected. Some other facets are also presented in this article. Additionally, the paper also studies a special case of the problem. Consider the following two operations:

- Replace a pair of parallel edges with a single edge between the same endpoints.
- Replace a pair of edges adjacent to a common node with degree 2 with a single edge between the endpoints other than the common node.

If applying these two operations consecutively until no more can be applied yields a graph with two nodes and a single edge between them, i.e., a complete graph with two nodes, then the graph is said to be series-parallel [34]. It is shown that the polytope is completely described by the trivial and cut inequalities if the underlying graph is series-parallel. Consequently, the 2-edge connected subgraph problem can be solved in polynomial time in this case.

A similar case arises if the underlying graph is a Halin graph. A Halin graph is a planar graph constructed from a plane embedding of a tree with at least four vertices and with no vertices of degree 2, by connecting all the leaves of the tree (the vertices of degree 1) with a cycle that passes around the tree [10]. Barahona and Mahjoub [6] consider the 2-edge and 2-node connected subgraph problem on

Halin graphs and complete descriptions of the polytopes are provided. If the 2-edge connected polytope is completely described by the cut and trivial inequalities for a graph, then the graph is called *Perfectly 2-edge connected*. Mahjoub [35] introduces new classes of Perfectly 2-edge connected graphs in addition to the Halin and series-parallel graphs and states sufficient conditions for a graph to be perfectly 2-edge connected.

A more detailed polyhedral analysis on the survivable network design problem could be found in [26]. In this article, Kerivin and Mahjoub generalize some known results from the literature and study in depth the problem on special graphs, such as series-parallel graphs. Polynomial time algorithms are proposed for the problem on such graphs. They also discuss the problem on graphs in which multiple edges are allowed.

Remember that there must be at least 2-edge disjoint paths between every node pair if the graph is to be 2-edge connected. This can be also interpreted as the connectivity requirement of a node, which is the lower bound on the number paths that must exist between the given node and the remaining nodes of the graph. Clearly, the problem is the 2-edge connected subgraph problem if the connectivity requirement of every node is 2. In some networks all nodes do not have the same importance and the connectivity requirement for the nodes that are less important can be relaxed and could be equal to 1 instead of 2. If the connectivity requirement of every node is 1 or 2, then the problem is referred to as *(1,2) survivable network design problem.* This problem is analyzed by Kerivin, Mahjoub and Nocq in [27]. The authors provide valid inequalities and conditions under which they are facet inducing. According to the polyhedral study, a branch and cut algorithm and computational results are presented. It is shown that some cut inequalities called F-partition inequalities are very effective in these problems. This inequality class is important as it can also be extended to our problems. Another class of valid inequalities, namely the partition inequalities, for the (1,2)-survivable network design polytope is described. If the connectivity requirement of every node is 2, then the partition inequalities are implied by other constraints, however, they are useful when some nodes have lower connectivity requirements. The separation algorithms of partition inequalities are also discussed by Kerivin and Mahjoub [24].

Vandenbussche and Nemhauser [42] studied the 2-edge connected subgraph problem on graphs in which multiple edges are allowed. The authors used the relation between this problem and the Graphical Traveling Salesman Problem (GTSP), in which the objective is to find a 2-edge connected subgraph such that every node has an even degree, to exploit the polytope of their problem. The authors discussed how facets of GTSP are modified to obtain facets for 2-edge connected subgraph problem.

Baïou and Mahjoub [3] analyze the Steiner 2-edge connected subgraph problem on series-parallel graphs. This problem differs from the 2-edge connected subgraph problem in the sense that there exists a set of nodes $S \subset V$ where V is the node set of the graph, and it is necessary to find a subgraph such that there exist at least 2-edge disjoint paths between every node pair u, v with $u \in S$ and $v \in S$. The associated polyhedra is analyzed and a complete formulation for the problem is provided since the authors focus on the problem defined on series-parallel graphs. Similarly, Baïou and Correa [2] are interested in a slightly different version. In this study the aim is to find a 2-edge connected subgraph, which is not necessarily spanning, with minimum cost. This problem can be seen as a generalization of the Steiner 2-edge connected subgraph problem. There are costs associated with the nodes and edges in this problem. The nodes that will be on the 2-edge connected subgraph are determined based on the weights of the nodes. Linear relaxations and facets are presented in this article. Separation problems for the facets are also discussed.

In survivable network design, it may be required to have a network such that each edge of the network belongs to a cycle with length less than or equal to some positive $K \geq 3$. These cycles are referred to as *bounded rings* and the problem is then called *survivable network with bounded rings problem.* If the bounds on the length of the rings are not imposed then the optimal solutions usually consist of a large cycle (see [15]). Although this makes it possible to reroute the signals in case of a failure, the signal may have to travel a very long distance. The bounded rings ensure that in case of a failure, the data could be routed without having to travel too long. There are various problem types which are given names according to the structure of the survivable network. Studies on the bounded ring problems on networks with different architectures can also be found in the literature.

Fortz et al. [14] studied the 2-connected networks with bounded meshes problem. The network to be designed is 2-node connected and every edge of the network is included in a cycle whose length is bounded with some constant K. Some valid inequalities are proposed and corresponding separation routines are discussed. As an exact solution methodology, a branch and cut algorithm is developed. However, a heuristic algorithm is also proposed by the authors since the size of the problems that could be solved optimally is small. Computational results are presented for both algorithms.

Fortz and Labbe [13] analyze the same problem and they propose a new formulation. Some facet defining inequalities are derived. The separation algorithms for them are discussed and a branch-and-cut algorithm is developed for the solution of this problem.

Fortz et al. [15] study the two-edge connected subgraph with bounded rings problem. In this problem, the aim is to design a minimum cost two edge connected network in which every edge belong to a cycle with length less than or equal to some positive constant K. A mathematical formulation is proposed by the authors and the associated polytope is analyzed. Some valid inequalities, called cycle inequalities, for the problem are identified and the conditions under which the inequalities are facet defining are described. The separation procedures are studied and it is shown that the inequalities can be separated in polynomial time provided that $K \leq 4$. The authors propose a branch-and-cut algorithm and provide computational results in the article.

In a similar problem type, the length of the paths rather than the length of the cycles are restricted. In other words, *k-edge disjoint hop constrained paths problem* is defined as designing a minimum cost network in which there exist k edge disjoint paths with lengths less than some positive K between every node pair. In such applications , the length of a path is the number of edges (hops) in the particular path. Dahl et al. [11] analyzed the polytope associated with k-edge disjoint 2-hop constrained paths problem. The length of the paths are bounded with 2 in their problem. The authors present a formulation for the problem and discuss the characteristics of the polytope.

As it can be seen from the studies discussed above both levels of our problem have been studied in the literature individually. However, in 2ECSSP and 2ECSDHP both levels have to be considered simultaneously to find the optimal solution. There are some articles, which focus on two-level network design problems, in the literature, however, the problems are decomposed into two subproblems in most of the papers. Klincewicz [29] present a survey for two-level network design problems. Star/star network design problem is one of the special cases of the network design problem. In this problem, both the backbone and the access networks are star networks. This problem is equivalent to the *capacitated facility location problem* or *uncapacitated facility location problem* depending on the capacity structure of the network. Facility location problem is widely studied in the literature. Tree/star problem is another variation, in which the backbone network is a tree and the access networks are star graphs. Although not widely studied, articles interested in this problem could be found in the literature. This problem is solved in two phases or with heuristics.

Fully interconnected/star networks are of great interest, because a complete backbone network provides the highest level of survivability. Besides, this problem is also related to the facility location and p-median problems. Therefore, many studies on this problem can be found.

Although rare, network design problems where the access networks are trees are also studied [29]. Star/tree, tree/tree and path/tree network design problems are some of the examples.

The last type of problem we consider is about design of a ring/star network, in which the backbone network is a cycle and the access nodes are connected directly to hubs to form star networks. This problem is closely related to our problem because a cycle or ring is a special type of 2-edge connected networks where degree of each node is equal to 2. An exact solution method to the ring/star problem (RSP) is proposed by Labbe et al. [30]. They solved both levels, selection of hubs and design of the networks, simultaneously. The polytope associated with the RSP is analyzed and some facets of the polytope are identified. Based on these facets a branch-and-cut algorithm is developed and computational results are presented. In addition, the authors devise heuristic methods for this problem. Although the RSP and our problem, 2ECSSP, are similar, there are differences in the polyhedral aspect. While the RSP polytope is close to the TSP polytope, the one associated with our problem is related to 2-edge connected subgraph polyhedron.

Although both sub-problems, namely network design and hub location problems are widely studied, there are few studies that consider both simultaneously. Solving the problems in two stages separately will result in sub-optimal solutions since the problems affect each other. Besides, it is observed that 2ECSSP has not been analyzed in the literature until our study. Therefore in this study, our main contribution is to propose an exact solution method which solves both parts of the problem at the same time. Our contribution to the literature, Fouilhoux et al. [16] focuses on the 2ECSSP and proposes a solution methodology to the problem.

As we stated in Chapter 1, the rapid increase in the amount of traffic on the telecommunications networks makes the use of optical networks, which can offer high transmission capacities, necessary. The regenerator placement problem can be found in the literature. But, we first give some literature on the optical networks so that some concepts related to them can be explained.

Shen and Grover [39] classify optical networks into three main classes according to the regeneration functions of their nodes. A network is referred to as *transparent* if its nodes do not have any regeneration function. When regeneration is available at every node, the network is called an *opaque* network. *Translucent* networks lie between these two extreme cases, i.e. regeneration is available only at some nodes of the network. Similarly a regenerator node is called *opaque* while other nodes are called *transparent.* Signal routes are referred to as *lightpaths* and the segments of a lightpath residing between two consecutive regeneration points, i.e., regenerators and source/destination nodes, are referred to as *transparent segments* in translucent networks. There is strong interest in implementing translucent networks due to significant cost savings because, it has been observed that, on the average, 20% of regeneration nodes are sufficient to achieve a performance close to that of an opaque network (see [46]). However, placing the minimum number of regenerators on the network such that the performance of an opaque network can be achieved is a challenging problem (see [28]).

The importance of survivability increases as the amount of data transmitted through a link increases since a failure would result in significant losses. [5, 23, 39, 41] discuss different ways of signal recovery and [23] states that path restoration, which is actually utilizing a pre-determined disjoint route in case of failures, is more effective if the reliability of links and nodes are not too different. Besides, in some special cases, *1+1 protection architecture*, in which the same data is transferred through both paths simultaneously, is employed in order to recover from the failure very quickly (see [19]). Although network survivability improves as the number of disjoint paths between source/destination nodes in the network increases, it has been shown in [20] that networks, in which there are at least two edge disjoint paths between each pair of nodes, are cost effective and provide an adequate level of survivability. This is consistent with our choice of 2-edge connected architecture and shows that the choice is reasonable.

Like the 2-edge connected subgraph problem, the problem of finding edge-disjoint paths has also been widely researched in the literature. For example, [43] shows the NP-completeness of some edge-disjoint path problems and [33] shows that the problem of finding two edge-disjoint paths such that length of the longer one is minimized is strongly NP-complete. The latter is of great interest to us, since solving this problem reveals the node pairs which can communicate without regenerators.

RPP has been introduced and addressed in [49] where the authors propose two heuristic algorithms for minimizing the number of regenerators. It is assumed in that study that paths should be simple. [39] employs a different approach, namely segment-based survivability for optical signal recovery, in which opaque nodes are used to detect failures. They propose a heuristic algorithm to solve the regenerator placement problem and find the optimal solution by complete enumeration.

In optical networks, multiple signals can be transmitted through a link simultaneously since there are different wavelengths assigned to each signal. If at least two signals using the same wavelength meet at a node, then the signals are blocked. Therefore, routes and wavelengths of the lightpaths should be chosen so that these blocking events are minimized, which is known as the Routing and Wavelength Assignment Problem. Wavelength Assignment problem is also studied in the literature together with the regenerator placement problem. [48] considers this problem and does not ignore the need for regeneration, however the objective is to minimize the blocking probability of the optical signals rather than minimizing the number of regenerators to be installed on the network. The nodes to be used as regenerators are determined according to the blocking probability of the signals due to wavelength unavailability. After determining the places of the regenerator nodes, routing algorithms are employed. [28] studies a similar problem and uses two approaches to the problem, the first one being a minimal-cost placement which minimizes the blocking of lightpaths using dynamic programming, the second being a heuristic for locating the signal regeneration nodes. A comparison between these two algorithms and others proposed in different studies is also given. As it can be seen although the problem of Wavelength Assignment and Regenerator Placement are considered together in these articles, they are either solved separately or heuristically. In RPP, we are not going to take the Wavelength Assignment Problem into account.

In [46] and [47], regenerator placement problem is solved first and routes are determined after regenerator places are selected. The problem is solved using heuristic methods and the performance of the system is evaluated via simulation models. Sparse regeneration is assumed in these papers, i.e. the signals are traversed as long as possible before regeneration is inevitable. [7] aims to predict the probability of regeneration needs at each node to determine the nodes on which regenerators should be placed. None of these studies consider network survivability.

The necessity for the optical networks in order to handle the rapidly increasing data traffic, the requirement of data regeneration and the question of why minimization of the number regenerators is an important are discussed in [8, 36]. There are different forms of signal regeneration in optical networks. Although we will not go into much detail, some preliminary information will be useful to understand why regeneration in optical networks differs from the regeneration in traditional electrical networks. Three methods of regeneration can be described as follows:

1R Regeneration

2R Regeneration and Reshaping

3R Regeneration, Reshaping and Retiming

The simplest way of regeneration is 1R regeneration. However, the current optical networks use the fiber optic cables only as a transmission medium. Therefore, 1R regeneration may not be applicable in all cases, making the use of 2R and 3R regeneration necessary. The 2R and 3R regenerations first convert the optical signals to electrical signals and then regenerates the optical signals with reshaping and possibly retiming. This conversion makes the signal regeneration in optical networks complicated. Therefore the minimization of regenerators becomes an important objective. The details of regeneration methods can be found in [8, 36].

The RPP is also studied in a recent article of Chen et al. [9]. The authors consider the RPP without the survivability requirement i.e., they assume only one path between every node pair is sufficient. They search locations for the regenerators and try to minimize the number of regenerators installed on the network. In the article the authors show that the problem is NP-Hard and propose heuristic methods together with a branch-and-cut algorithm. Some preprocessing methods are proposed which are valid when only a working path is considered.

Based on this literature review, it can be seen that problems that are similar to the three problems we are considering, have been studied commonly in the literature. It is also observed that there are many application areas of these problems. This shows that our problems are of great interest both theoretically and practically. In addition, we can also note that the versions of the problems we propose have not been tackled in the literature. Our study provides the analysis of the problems and their structures. The exact solution methodologies provided for the problems also bring in a practical perspective to our study. Therefore, we believe our study will make a significant contribution to the literature in the telecommunications context.

Chapter 3

Hierarchical Survivable Network Design with Single Homing

We first analyze the 2ECSSP, which is the problem of designing a two level telecommunications network with a survivable backbone component. As we described earlier, the network consists of two types of devices namely, hubs and users. Similarly the network is mainly composed of two types of networks. The first component is called the backbone network and connects the hubs. The second level is the access network, which connects the users to the hubs in the backbone network. Since the data is mainly transmitted via the backbone network, the survivability of the backbone is more important. Therefore, we first focus on the survivability of the backbone.

To achieve survivability, we want the backbone to be 2-edge connected. Each user is assigned to one hub so that the users are connected to the backbone, which means the local access networks have star architecture. In this problem we need to determine the number and location of the hubs among a set of nodes, and the remaining nodes will form the set of users. We consider the cost of installing links in the backbone and the cost of connecting users to hubs. The objective in this problem is to find the design that satisfies the survivability requirements and has minimum cost.

Before formally defining the problem, we want to present some of the notation we are going to use in this chapter. Here we provide only a part of the notation which is commonly used and the remaining notation will be given as needed. We consider directed and undirected graphs. We denote an undirected graph by $G = (V, E)$ where V is the node set and E is the edge set of G. If $e \in E$ is an edge between two nodes i and j, then we also write $e = ij$ or $e = \{i, j\}$ to denote e. If a node i is one of the endpoints of an edge e we say $i \in e$, and $i \notin e$, otherwise. If V_1 and V_2 are two node subsets such that $V_1 \cap V_2 = \emptyset$, then we denote by $[V_1, V_2]$ the set of edges having one node in V_1 and the other in V_2. Given a set $S \subseteq V$, we let $\delta_G(S) = [S, V \setminus S]$, that is the set of edges having exactly one node in S. We will omit the subscript if the context is clear. The edge set $\delta(S)$ is called a cut. For $i \in V$, we will write $\delta(i)$ instead of $\delta(\{i\})$.

A directed graph will be denoted by $D = (V, A)$ where V is the node set and A is the arc set. If $a \in A$ is an arc from node i to node j, then we also write $a = (i, j)$ to denote a. For $S \subseteq V$, we let $G(S)$ $(D(S))$ denote the subgraph of G (D) induced by S, that is the subgraph whose node set is S and edge (arc) set is $E(S)$ $(A(S))$, the set of edges (arcs) in G (D) having both nodes in S.

Given a vector $x \in \mathbb{R}^{|E|}$ and $F \subseteq E$, we let $x(F) = \sum_{e \in F} x_e$.

We now can give the formal description of the 2ECSSP. We consider an undirected graph, $G = (V, E)$, and a directed graph, $D = (V, A)$, simultaneously. The undirected graph is used for the backbone while we use the directed one to define the local access networks. Both graphs have the same node set $V = \{0, 1, \ldots, n\}$ which defines the set of terminals. Node 0 is a special concentrator corresponding to the root node in the two level network infrastructure. We assume that this node is always a concentrator. Let $E = \{\{i, j\} : i \in V, j \in V \setminus \{i\}\}$ be the set of undirected edges representing the set of potential backbone links. Thus we assume a complete graph in terms of edges, i.e., any node pair can be connected directly in the backbone network. Let $A = \{(i, j) : i \in V, j \in V\}$ be the set of directed arcs which are used to represent the assignments of users to the hubs. Note that we also assume a complete graph for the directed arcs. In addition, A includes the loops, which will be used to indicate a node is assigned to itself meaning that it is a hub. This set will be revised later. We associate a fixed setup cost of installing a backbone link c_e with each edge $e \in E$. Similarly, there is an assignment cost of d_{ij} associated with assigning terminal $i \in V$ to concentrator $j \in V$. In particular, d_{ii} corresponds to the cost of installing a concentrator at node $i \in V$. Note that d_{ij} and d_{ji} might be different. We assume c_{ij} and d_{ij} are nonnegative.

Given the node set V, 2ECSSP seeks a partition of V into C and T such that $0 \in C$. A set of backbone links $E' \subseteq E$ between nodes in C is chosen such that the graph (C, E') is 2-edge connected. Finally, each node in T is assigned to one in C such that the total cost of installing backbone links and concentrators and assigning terminals to concentrators is minimum. 2ECSSP is NP-hard since it possesses as a special case the 2-edge connected subgraph problem, which is NP-hard [35].

3.1 Mathematical Formulation

We propose an integer linear program for the 2ECSSP. First we define the following decision variables:

$$x_e = \begin{cases} 1, & \text{if } e \text{ is used in the backbone network} \\ 0, & \text{otherwise} \end{cases}$$

$$y_{ij} = \begin{cases} 1, & \text{if } i \text{ is assigned to node } j \\ 0, & \text{otherwise} \end{cases}$$

If a concentrator is installed at node $i \in V$ then node i is assigned to itself, i.e., $y_{ii} = 1$.

Using these two sets of binary variables, we can model the 2ECSSP as follows:

$$z = \min \sum_{e \in E} c_e x_e + \sum_{i \in V} \sum_{j \in V} d_{ij} y_{ij} \tag{3.1}$$

s.t.

$$\sum_{j \in V} y_{ij} = 1 \qquad \forall i \in V, \tag{3.2}$$

$$x_{ij} + y_{ij} \leq y_{jj} \qquad \forall (i,j) \in A,\ i \neq j, \tag{3.3}$$

$$x(\delta(S)) \geq 2 \sum_{j \in S} y_{ij} \qquad \forall S \subseteq V \setminus \{0\}, i \in S, \tag{3.4}$$

$$y_{00} = 1 \tag{3.5}$$

$$x_e \in \{0,1\} \qquad \forall e \in E, \tag{3.6}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in A. \tag{3.7}$$

The first and second terms of the objective function (3.1) denote the cost of backbone and local access networks, respectively. Constraint (3.2) is the assignment constraint which implies that either a concentrator is installed at a node or that node is assigned to another concentrator. Constraint (3.3) is used to define the relation between the edges, arcs and concentrators. If an edge is used in the backbone then concentrators are installed at both endpoints of this edge. Similarly, if a node i is assigned to node j then a concentrator must be installed at j. Constraints (3.4) are called the cut constraints and ensure that the backbone network is 2-edge connected. To satisfy 2-edge connectivity we need to install at least two links between two node sets in which there is at least one concentrator. Consider the node subset $S \subseteq V \setminus \{0\}$ and a node $i \in S$ depicted in Figure 3.1. Due to the root node we know that there is at least one concentrator in $V \setminus S$. If i is assigned to some node in set S, i.e., if $\sum_{j \in S} y_{ij} = 1$, then there is at least one concentrator, say k in S, implying that i and k must be linked by at least two edge-disjoint paths, and hence at least two edges from $\delta(S)$ have to be included in the backbone network. Note that the cut constraints are defined by a node set S and a fixed node i from S. Actually i can be chosen from $V \setminus S$, but it can be shown that in that case the cut inequalities become redundant. Therefore, they are not included in the mathematical model. Constraint (3.5) fixes the value of y_{00} to one and hence a concentrator is installed at the root node 0. Finally, (3.6)

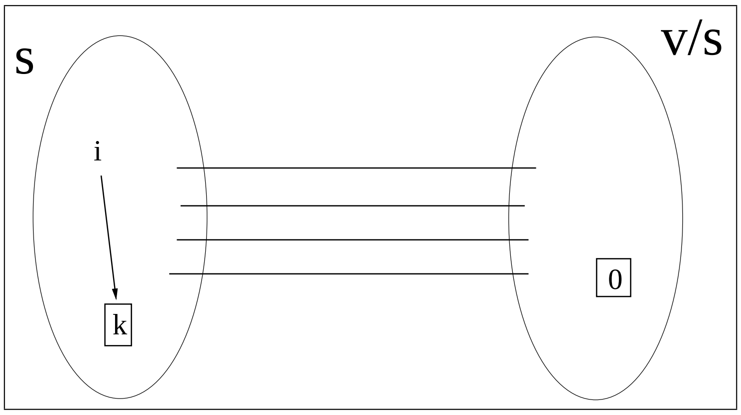

Figure 3.1: Cut inequalities

and (3.7) are the integrality constraints.

2ECSSP is a relaxation of the ring/star network design problem and the formulation we propose is obtained by removing the degree constraints from the formulation of the ring/star network design problem given in Labbé et al. [30]. With this modification, we allow the backbone network to have not only ring structure but also other 2-edge connected architectures.

Our model is based on the assumption of the existence of a root node which is always a concentrator. This root node might be a central unit to which other concentrators should be connected or it might be desired to connect the backbone network to an already existing higher level network at this point. In such cases, the existence assumption of a root node is reasonable. However, this assumption is not restrictive and if there is no such node, then the cut constraints (3.4) can be modified as follows:

$$x(\delta(S)) + 2\sum_{j\in V\setminus S} y_{ij} + 2\sum_{j\in S} y_{kj} \geq 2 \qquad \forall S \subset V, \forall i \in V, \forall k \in V\setminus\{i\}$$

These constraints force the model to install at least two edges between sets S and $V \setminus S$ if at least one concentrator is installed in each set.

3.2 Polyhedral Analysis

As the proposed integer formulation has exponential number of constraints, it is not possible to solve the model directly even for medium sized models. For this reason, a branch-and-cut algorithm is required to solve the model optimally. In this section, we present a polyhedral analysis for the convex hull of the solutions to the 2ECSSP, and this information will be used to develop the branch-and-cut algorithm. Before performing the analysis we provide some basic information that will be used used frequently.

3.2.1 Preliminaries

In this section we discuss some preliminaries for the polyhedral analysis. For detailed information one can refer to the books of Wolsey[44] and Nemhauser and Wolsey [37]. Since we are interested in the description of a polyhedron associated with a mathematical program, we first define which inequalities are redundant in the description.

Definition 3.1 *If $\pi x \leq \pi_0$ and $\mu x \leq \mu_0$ are two valid inequalities for $\mathcal{P} \subseteq R^n_+, \pi x \leq \pi_0$ dominates $\mu x \leq \mu_0$ if there exists $u > 0$ such that $\pi \geq u\mu$ and $\pi_0 \leq u\mu_0$, and $(\pi, \pi_0) \neq (u\mu, u\mu_0)$ [37].*

Definition 3.2 *A valid inequality $\pi x \leq \pi_0$ is redundant in the description $\mathcal{P}$, if there exists $k \geq 2$ valid inequalities $\pi^i x \leq \pi^i_0$ for $i = 1, \ldots, k$ for $\mathcal{P}$, and weights $u_i > 0$ for $i = 1, \ldots, k$ such that $(\sum_{i=1}^k u_i \pi^i)x \leq (\sum_{i=1}^k u_i \pi^i_0)$ dominates $\pi x \leq \pi_0$ [37].*

According to Definitions 3.1 and 3.2 we can say that an inequality is not necessary for the description of a polyhedron if it is implied by other inequalities since even if that inequality is removed from the description, the polyhedron remains the same. However, it is not always easy to see if an inequality is redundant, so we need other tools to identify whether an inequality is necessary for the description of the polyhedra or not. Before defining the properties of necessary inequalities, we give a few additional definitions.

Definition 3.3 *The points $x^1, \ldots, x^k \in R^n$ are affinely independent if the $k-1$ directions $x^2 - x^1, \ldots, x^k - x^1$ are linearly independent [37].*

Note that linear independence implies affine independence, however, the converse is not true.

Definition 3.4 *The dimension of $\mathcal{P}$, denoted $dim(\mathcal{P})$, is one less than the maximum number of affinely independent points in $\mathcal{P}$ [37].*

According to this definition, $\mathcal{P} \subseteq R^n$ is full-dimensional, i.e. $dim(\mathcal{P}) = n$, if and only if there are $n+1$ affinely independent points in $\mathcal{P}$.

Definition 3.5 *$\mathcal{F}$ defines a face of the polyhedron $\mathcal{P}$ if $\mathcal{F} = \{x \in \mathcal{P} | \pi x = \pi_0\}$ for some valid inequality $\pi x \leq \pi_0$ of $\mathcal{P}$. $\mathcal{F}$ is said to be a proper face if $\mathcal{F} \neq \mathcal{P}$ and $\mathcal{F} \neq \emptyset$ [37].*

Definition 3.6 *A face $\mathcal{F}$ of $\mathcal{P}$ is a facet of $\mathcal{P}$ if $dim(\mathcal{F}) = dim(\mathcal{P}) - 1$ [37].*

If $\mathcal{F}$ is a facet of $\mathcal{P}$, the valid inequality associated with $\mathcal{F}$ is referred to as *facet defining* or *facet inducing* inequality. Facets are important for the description of polyhedra.

Proposition 3.1 *For each facet $\mathcal{F}$ of $\mathcal{P}$, one of the inequalities representing $\mathcal{F}$ is necessary in the description of $\mathcal{P}$ [37].*

Proposition 3.2 *Every inequality $ax \leq b$, that represents a face of $\mathcal{P}$ of dimension less than $dim(\mathcal{P}) - 1$ is irrelevant to the description of $\mathcal{P}$ [37].*

There are several methods that can be used to show that an inequality is facet defining. The first one is the direct method. Suppose $\mathcal{F}$ is a face of polyhedron $\mathcal{P}$ with $dim(\mathcal{P}) = n$. If n affinely independent solutions can be found in $\mathcal{F}$, then $\mathcal{F}$ is a facet of $\mathcal{P}$. The second is an indirect method, which is referred to as *maximality method*. We will give the theorem that states the method formally, however, we first want to explain the idea behind the method. Consider polyhedron $\mathcal{P} = \{x \in R^n | Ax \leq b\}$. Let $A^= x = b^=$ be the equations that are satisfied by every point $x \in \mathcal{P}$. Then the relation between the dimension of a polyhedron and the rank of the matrix $(A^=, b^=)$ is established as follows.

Proposition 3.3 *If $\mathcal{P} \subseteq R^n$, then $dim(\mathcal{P}) + rank(A^=, b^=) = n$ [37].*

For simplicity, it is assumed that the polyhedron under consideration is full dimensional, however this assumption can be easily relaxed. Now consider the face $\mathcal{F}$ of $\mathcal{P}$ induced by valid inequality $\pi x \leq \pi_0$. We know that $dim(\mathcal{F}) \leq n-1$, as there is at least one equation satisfied by all points in $\mathcal{F}$. If there exists another equation that is satisfied by all $x \in \mathcal{F}$, then $\mathcal{F}$ is not facet defining unless the equalities are linearly dependent, i.e., the second is a multiple of the first one. This result is formally stated in the following theorem.

Theorem 3.1 *Let $\mathcal{F} = \{x \in \mathcal{P} | \pi x = \pi_0\}$ be a proper face of $\mathcal{P} \subseteq R^n$. The following two statements are equivalent:*

1. *$\mathcal{F}$ is a facet of $\mathcal{P}$.*
2. *If $\lambda x = \lambda_0$ for all $x \in \mathcal{F}$ then $(\lambda, \lambda_0) = (\alpha\pi, \alpha\pi_0)$ for some $\alpha \in R$.*

[37]

3.2.2 A new formulation

Using Proposition 3.3, it can be seen that the polytope defined by the constraints of the mathematical model is not full dimensional as there are some equality constraints such as the assignment constraints. This will make the polyhedral analysis harder. So in order to obtain a full dimensional polytope we make some modifications on the formulation. Note that, there will not be any changes in the problem definition or in the associated polytope. We will only obtain a different representation of the polyhedron.

We start with projecting out the variables y_{ii}, corresponding to the loops in the arc set. Using the assignment constraints (3.2), for $i \in V \setminus \{0\}$, we can eliminate variable y_{ii} by substituting $y_{ii} = 1 - \sum_{j \in V \setminus \{i\}} y_{ij}$. Besides, the values of the variables related to assignment of the root node are known a priori. We know that $y_{00} = 1$ and $y_{0i} = 0$ for all $i \in V \setminus \{0\}$. Therefore, these variables can also be dropped from the formulation.

Note that a constraint of type (3.3) is defined by an arc $(i, j) \in A$ such that $i \neq j$. As values of some variables corresponding to certain arcs, i.e., the arcs emanating from the root node, the substitution yields three cases for these inequalities. Consider an arc $(i, j) \in A$ such that $i \neq j$. If 0 is not one of the nodes (i, j), we have $x_{ij} + y_{ij} + \sum_{k \in V \setminus \{j\}} y_{jk} \leq 1$. On the other hand, if $i = 0$, we get $x_{0j} + y_{0j} + \sum_{k \in V \setminus \{j\}} y_{jk} \leq 1$, and if $j = 0$ we obtain $x_{i0} + y_{i0} + \sum_{k \in V \setminus \{0\}} y_{0k} \leq 1$. Replacing $y_{00} = 1$ and $y_{0i} = 0$ for all $i \in V \setminus \{0\}$, we obtain the following inequalities.

$$x_{ij} + y_{ij} + \sum_{k \in V \setminus \{j\}} y_{jk} \leq 1 \qquad (i, j) \in A : i \neq 0, j \neq 0, i \neq j$$

$$x_{0i} + \sum_{k \in V \setminus \{i\}} y_{ik} \leq 1 \qquad (0, i) \in A : i \neq 0 \tag{3.8}$$

$$x_{0i} + y_{i0} \leq 1 \qquad (i, 0) \in A : i \neq 0 \tag{3.9}$$

Clearly (3.8) dominates (3.9), so we remove the dominated one from the formulation.

As substitution eliminates some variables the arc set needs to be redefined as $A = \{(i, j) : i \in V \setminus \{0\}, j \in V \setminus \{i\}\}$. We also need to modify the assignment costs as $d'_{ij} = d_{ij} - d_{ii}$ for each $(i, j) \in A$. Now we can present our new equivalent formulation which is used for polyhedral analysis.

$$z = \sum_{i \in V} d_{ii} + \min \sum_{e \in E} c_e x_e + \sum_{(i,j) \in A} d'_{ij} y_{ij}$$

s.t.

$$x_{ij} + y_{ij} + \sum_{k \in V \setminus \{j\}} y_{jk} \leq 1 \qquad \forall (i,j) \in A : j \neq 0 \tag{3.10}$$

$$x_{0i} + \sum_{k \in V \setminus \{i\}} y_{ik} \leq 1 \qquad \forall i \in V \setminus \{0\} \tag{3.11}$$

$$x(\delta(S)) + 2 \sum_{j \in V \setminus S} y_{ij} \geq 2 \qquad \forall S \subseteq V \setminus \{0\}, i \in S \tag{3.12}$$

$$0 \leq x_e \leq 1 \qquad \forall e \in E \tag{3.13}$$

$$0 \leq y_{ij} \leq 1 \qquad \forall (i,j) \in A. \tag{3.14}$$

$$x_e \text{ integer} \qquad \forall e \in E \tag{3.15}$$

$$y_{ij} \text{ integer} \qquad \forall (i,j) \in A. \tag{3.16}$$

Inequalities (3.10)-(3.11) will be called *clique inequalities*, inequalities (3.12) will be called *cut inequalities* and inequalities (3.13)-(3.14) are called *trivial inequalities*. Let $X = \{(x,y) \in R^{|E|+|A|} : (x,y) \text{ satisfies (3.10)-(3.16)}\}$ and $\mathcal{P} = conv(X)$.

Having defined our polytope, we can present the polyhedral analysis. In this chapter we show that the constraints of the integer linear program define facets of $\mathcal{P}$. In addition we present some valid inequalities and provide some conditions under which they define facets. We also study the relationship between some facets of a special stable set polytope and the facets of $\mathcal{P}$. But before the polyhedral analysis, we introduce some more notation. For $e \in E$, let χ_e be a unit vector of size $|E|$ with the entry corresponding to edge e equal to 1 and other entries equal to 0. Similarly, for $(i,j) \in A$, let γ_{ij} be a unit vector of size $|A|$ with the entry corresponding to arc (i,j) equal to 1 and other entries equal to 0. Thus, for a vector $x \in \{0,1\}^{|E|}$ (resp. $y \in \{0,1\}^{|A|}$), if F is the set of edges e (resp. arcs a) such that $x_e = 1$ (resp. $y_a = 1$), then x (resp. y) can also be written as $\sum_{e \in F} \chi_e$ (resp. $\sum_{a \in F} \gamma_a$).

Hereafter we assume that $|V| \geq 5$. If $|V| < 5$, some of the inequalities of the formulation do not define facets of the associated polytope, and some special facet defining inequalities appear due to the small size of the graph. However, we do not go into the details as the problem is easy to solve in that case.

We start the analysis by determining the dimension of $\mathcal{P}$.

Theorem 3.2 *$\mathcal{P}$ is full dimensional.*

Proof Consider the solutions $(\sum_{e \in E} \chi_e, 0)$, $(\sum_{e \in E \setminus \{e'\}} \chi_e, 0)$ for $e' \in E$ and $(\sum_{e \in E \setminus \delta(i)} \chi_e, \gamma_{ij})$ for $(i,j) \in A$. They are in $\mathcal{P}$ and are affinely independent, and hence $dim(\mathcal{P}) = |E| + |A|$. $\Box$

3.2.3 Basic Inequalities

Knowing the dimension of $\mathcal{P}$, we can study the facet defining inequalities. With the term basic inequalities we refer to the constraints of the formulation. We start by analyzing the trivial inequalities.

Theorem 3.3 *For $e \in E$, inequality $x_e \geq 0$ is facet defining for $\mathcal{P}$.*

Proof Let $\mathcal{F} = \{(x,y) \in \mathcal{P} : x_e = 0\}$. The solutions $(\sum_{e' \in E \setminus \{e\}} \chi_{e'}, 0)$, $(\sum_{e'' \in E \setminus \{e,e'\}} \chi_{e''}, 0)$ for $e' \in E \setminus \{e\}$, $(\sum_{e' \in E \setminus \delta(k)} \chi_{e'}, \gamma_{kl})$ for $(k,l) \in A$ with $k \in e$ (k is an endpoint of e), and $(\sum_{e' \in E \setminus (\delta(k) \cup \{e\})} \chi_{e'}, \gamma_{kl})$ for $(k,l) \in A$ with $k \notin e$ (k is not an endpoint of e), constitute a family of $|E| + |A|$ affinely independent solutions in $\mathcal{F}$. $\Box$

Theorem 3.4 *For $(i,j) \in A$, inequality $y_{ij} \geq 0$ is facet defining for $\mathcal{P}$.*

Proof Let $\mathcal{F} = \{(x,y) \in \mathcal{P} : y_{ij} = 0\}$. The solutions $(\sum_{e \in E} \chi_e, 0)$, $(\sum_{e \in E \setminus \{e'\}} \chi_e, 0)$ for $e' \in E$ and $(\sum_{e \in E \setminus \delta(k)} \chi_e, \gamma_{kl})$ for $(k,l) \in A \setminus \{(i,j)\}$ are in $\mathcal{F}$ and are affinely independent. $\Box$

Inequalities $x_{ij} \leq 1$ and $y_{ij} \leq 1$ are not facet defining as they are implied by constraints (3.10) and (3.11).

We now focus on the cut constraints (3.12) which are used to make the backbone network 2-edge connected. We provide necessary and sufficient conditions for these inequalities to be facet defining for $\mathcal{P}$ in the following theorem.

Theorem 3.5 *Let $S \subseteq V \setminus \{0\}$ such that $S \neq \emptyset$ and $i \in S$. Inequality (3.12) defines a facet of $\mathcal{P}$ if and only if $|S| \neq 2$ and $|V \setminus S| \neq 2$.*

Proof Suppose that $|S| = 2$ and $S = \{i, j\}$. Then inequality (3.12) for this choice of S and i is

$$x(\delta(\{i,j\})) + 2 \sum_{k \in V \setminus \{i,j\}} y_{ik} \geq 2. \tag{3.17}$$

Summing the cut inequalities (3.12) for $S = \{i\}$ and $S = \{j\}$ yields $x(\delta(i)) + x(\delta(j)) + 2\sum_{k\in V\setminus\{i\}} y_{ik} + 2\sum_{k\in V\setminus\{j\}} y_{jk} \geq 4$. Substituting $x(\delta(i)) + x(\delta(j)) = x(\delta(\{i,j\})) + 2x_{ij}$ and adding constraint (3.10), $-2x_{ij} - 2y_{ij} - 2\sum_{k\in V\setminus\{j\}} y_{jk} \geq -2$, we obtain inequality (3.17), and hence (3.17) is not facet defining.

Now suppose that $V \setminus S = \{0, j\}$ and $j \in V\setminus\{0, i\}$. Let $(x, y) \in X$ be a solution which satisfies the corresponding cut inequality (3.12) at equality. If $x_{0j} = 1$, since x induces a 2-edge connected subgraph, we should have $\sum_{k\in S} x_{0k} \geq 1$ and $\sum_{k\in S} x_{jk} \geq 1$. As inequality (3.12) for $S = V \setminus \{0, j\}$ is tight for (x, y), it thus follows that $\sum_{k\in S} x_{0k} = 1$ (and $\sum_{k\in S} x_{jk} = 1$). If $x_{0j} = 0$, then one should have $\sum_{k\in S} x_{0k} = 2$, and therefore $\sum_{k\in S} x_{jk} = 0$. In both cases (x, y) satisfies $x_{0j} - \sum_{k\in S} x_{jk} = 0$. As this equation is not a multiple of $x(\delta(V \setminus \{0, j\})) + 2(y_{i0} + y_{ij}) = 2$, inequality (3.12) is not facet defining.

Now suppose that $|S| > 2$ and $|V \setminus S| > 2$. Notice that as G is complete, $G(S)$ and $G(V \setminus S)$ are 2-edge connected. Let $\mathcal{F} = \{(x, y) \in \mathcal{P} : x(\delta(S)) + 2\sum_{j\in V\setminus S} y_{ij} = 2\}$. Suppose that every solution (x, y) in $\mathcal{F}$ also satisfies $ax + by = \beta$. We will show that $ax + by = \beta$ is a multiple of $x(\delta(S)) + 2\sum_{j\in V\setminus S} y_{ij} = 2$.

Consider solution $(x, 0)$ where $x = \sum_{e\in E(S)\cup E(V\setminus S)} \chi_e + \chi_{e_1} + \chi_{e_2}$ and e_1 and

e_2 are any two edges in $\delta(S)$. Let $e' \in \delta(S) \setminus \{e_1, e_2\}$. As $(x, 0)$ and the solutions $(x + \chi_{e'} - \chi_{e_1}, 0)$ and $(x + \chi_{e'} - \chi_{e_2}, 0)$ are both in $\mathcal{F}$, we have $a_{e_1} = a_{e_2} = a_{e'}$. Therefore $a_{e'} = \sigma$ for all $e' \in \delta(S)$ for some $\sigma \in \mathbb{R}$.

Let $e' \in E(S)$ and let e_1, e_2 be two edges in $\delta(S)$ incident to the two endpoints of e' such that $e_1 \cap e_2 \cap e' = \emptyset$. Consider the solution $(x, 0)$ where $x = \sum_{e \in E(S) \cup E(V \setminus S)} \chi_e + \chi_{e_1} + \chi_{e_2}$. As $(x, 0)$ and the solution $(x - \chi_{e'}, 0)$ are both in $\mathcal{F}$, we have $a_{e'} = 0$. We can show similarly that $a_{e'} = 0$ for all $e' \in E(V \setminus S)$.

Let $j \in V \setminus \{i, 0\}$ and let e_1, e_2 be two edges in $\delta(S) \setminus \delta(j)$ with different endpoints in S if $j \in S$ and with different endpoints in $V \setminus S$ if $j \in V \setminus S$. Consider $(x, 0)$ where $x = \sum_{e \in E(S) \cup E(V \setminus S)} \chi_e + \chi_{e_1} + \chi_{e_2}$. This solution is in $\mathcal{F}$. Observe that, $(x - \sum_{e \in \delta(j)} x_e \chi_e, \gamma_{jk})$ is also in $\mathcal{F}$ for any $k \in V \setminus \{j\}$. As $a_e = 0$ for all $e \in E(S) \cup E(V \setminus S)$ we have $b_{jk} = 0$.

Similarly, the solution $(x, 0)$ where $x = \sum_{e \in E(S) \cup E(V \setminus S)} \chi_e + \chi_{e_1} + \chi_{e_2}$ and e_1 and e_2 are two edges in $\delta(S) \setminus \delta(i)$ is in $\mathcal{F}$. Let $k \in S \setminus \{i\}$. As the solution $(x - \sum_{e \in \delta(i)} x_e \chi_e, \gamma_{ik})$ is also in $\mathcal{F}$, we have $b_{ik} = 0$.

Let $j \in V \setminus S$ and consider $(x, 0)$ where $x = \sum_{e \in E(S) \cup E(V \setminus S)} \chi_e + \chi_{e_1} + \chi_{e_2}$ and e_1 and e_2 are two edges in $\delta(S)$. As $(x - \sum_{e \in E(S)} \chi_e - \chi_{e_1} - \chi_{e_2}, \sum_{k \in S} \gamma_{kj})$ is in $\mathcal{F}$, $b_{kj} = 0$ for every $k \neq i$, $a_e = 0$ for all $e \in E(S)$, and $a_{e_1} = a_{e_2} = \sigma$, we have $b_{ij} = 2\sigma$.

If $|S| = 1$ or $|V \setminus S| = 1$, computation of a and b is almost the same. The difference is that if $|S| = 1$, then $E(S) = \emptyset$, there is not a node $j \in S \setminus \{i\}$ and there is not an arc (i, j) with $j \in S$ as $S = \{i\}$. Similarly, if $|V \setminus S| = 1$, then $E(V \setminus S) = \emptyset$ and there is not a node $j \in V \setminus (S \cup \{0\})$. So we do not calculate the corresponding coefficients. Computation of other coefficients is still valid.

Therefore, $ax + by = \beta$ is a multiple of $x(\delta(S)) + 2\sum_{j \in V \setminus S} y_{ij} = 2$ and $\mathcal{F}$ is a facet of $\mathcal{P}$. $\square$

Now the only inequalities that are not analyzed yet in the formulation are the clique inequalities. We will show that clique inequalities also define facets for $\mathcal{P}$. However, instead of analyzing the clique inequalities directly, we aim to investigate the relation between $\mathcal{P}$ and the stable set relaxation polytope. This will be a more general result and indirectly give information about the clique inequalities.

3.2.4 Stable set relaxation and clique inequalities

Consider the set consisting of solutions satisfying the constraints of the 2EC-SSP formulation other than the cut constraints. The constraints indicate which variables cannot be positive simultaneously and they define a stable set problem.

Let $X_S = \{(x,y) \in R^{|E|+|A|} : (x,y)$ satisfies (3.10), (3.11), (3.15), and (3.16)$\}$ and $\mathcal{P}_\mathcal{S} = conv(X_S)$. The polytope $\mathcal{P}_\mathcal{S}$ is a stable set polytope. Note that a feasible solution of $\mathcal{P}$ is also a feasible solution of $\mathcal{P}_\mathcal{S}$. Note also that we have shown there exists sufficiently many affinely independent solutions in $\mathcal{P}$ to make $\mathcal{P}$ full-dimensional. Therefore, as $\mathcal{P} \subseteq \mathcal{P}_\mathcal{S}$ and $\mathcal{P}$ is full dimensional, $\mathcal{P}_\mathcal{S}$ is also full dimensional. Similar reasoning is also valid for the facet defining inequalities. Let $\alpha x + \beta y \leq \beta_0$ be a facet defining inequality for $\mathcal{P}$. If this inequality is valid for $\mathcal{P}_\mathcal{S}$, then it also defines a facet of $\mathcal{P}_\mathcal{S}$. This implies that the inequalities $x_e \geq 0$ for $e \in E$ and $y_{ij} \geq 0$ for $(i,j) \in A$ are facet defining for $\mathcal{P}_\mathcal{S}$. The trivial inequalities $x_e \leq 1$ for $e \in E$ and $y_{ij} \leq 1$ for $(i,j) \in A$ are implied by constraints (3.10) and (3.11) and hence do not define facets of $\mathcal{P}_\mathcal{S}$. Moreover as X_S is an independence system, if $\alpha x + \beta y \leq \beta_0$ is a nontrivial facet defining inequality for $\mathcal{P}_\mathcal{S}$, then $\alpha \geq 0$, $\beta \geq 0$, and $\beta_0 > 0$.

In the following two theorems, we investigate how some of the facets of $\mathcal{P}_\mathcal{S}$ are related to those of $\mathcal{P}$ provided some conditions are satisfied.

Theorem 3.6 *Let $e = \{i,j\} \in E$ with $i \neq 0$ and $j \neq 0$. Suppose that inequality $\alpha_e x_e + \beta y \leq \beta_0$ is a nontrivial facet defining inequality for $\mathcal{P}_\mathcal{S}$. If*

i) for all $m \in V \setminus \{0,i,j\}$ and $l \in V \setminus \{0,i,j,m\}$, there exists a node $k \in V \setminus \{0,m,l\}$ such that $\beta_{km} = \beta_{kl} = \beta_{k0} = 0$, *ii) for $m \in V \setminus \{0, i, j\}$, there exists a node $k \in V \setminus \{0, i, j, m\}$ such that $\beta_{km} = \beta_{ki} = \beta_{kj} = \beta_{k0} = 0$,*

iii) for $m \in V \setminus \{0, i, j\}$, there exist two distinct nodes $k_1, k_2 \in V \setminus \{0, m\}$ such that $\beta_{k_1m} = \beta_{k_10} = \beta_{k_2m} = \beta_{k_20} = 0$ and $|\{k_1, k_2\} \cap \{i, j\}| \leq 1$,

iv) there exist two distinct nodes $k_1, k_2 \in V \setminus \{0, i, j\}$ such that $\beta_{k_10} = \beta_{k_1i} = \beta_{k_1j} = \beta_{k_20} = \beta_{k_2i} = \beta_{k_2j} = 0$,

are all satisfied, then the inequality $\alpha_e x_e + \beta y \leq \beta_0$ also defines a facet of $\mathcal{P}$.

Proof Let $e = \{i, j\} \in E$ with $i \neq 0$ and $j \neq 0$. Suppose that the inequality $\alpha_e x_e + \beta y \leq \beta_0$ is a nontrivial facet defining inequality for $\mathcal{P}_\mathcal{S}$ and that conditions *i)-iv)* are satisfied. Define $\mathcal{F}_\mathcal{S} = \{(x, y) \in P_S : \alpha_e x_e + \beta y = \beta_0\}$ and $\mathcal{F} = \{(x, y) \in P : \alpha_e x_e + \beta y = \beta_0\}$. Suppose that all solutions $(x, y) \in \mathcal{F}$ also satisfy $ax + by = b_0$. We will show that $ax + by = b_0$ is a multiple of $\alpha_e x_e + \beta y = \beta_0$. Let $e' = \{m, l\} \in E \setminus \{e\}$. There exists a solution $(x, y) \in \mathcal{F}_\mathcal{S}$ such that $x_{e'} = 1$. Let $V' = \{v \in V : \sum_{k \in V \setminus \{v\}} y_{vk} = 0\} \cup \{0\}$, i.e., V' is the set of backbone nodes.

Suppose that $m \in V \setminus \{0, i, j\}$ and $l \in V \setminus \{0, i, j, m\}$. As $x_{e'} = 1$, we know that $|V'| \geq 3$. If $|V'| = 3$, then $V' = \{0, m, l\}$. By *i)*, there exists $k \in V \setminus \{0, m, l\}$ such that $\beta_{km} = \beta_{kl} = \beta_{k0} = 0$. The solution (x', y') with $x' = x + \sum_{e'' \in E(V' \cup \{k\})}(1 - x_{e''})\chi_{e''}$ and $y' = y - y_{km}\gamma_{km} - y_{kl}\gamma_{kl} - y_{k0}\gamma_{k0}$ is in $\mathcal{F}$. Also the solution $(x' - \chi_{e'}, y')$ is in $\mathcal{F}$ and hence $a_{e'} = 0$. If $|V'| \geq 4$ then both solutions (x', y) with $x' = x + \sum_{e'' \in E(V') \setminus \{e\}}(1 - x_{e''})\chi_{e''}$ and $(x' - \chi_{e'}, y)$ are in $\mathcal{F}$ implying that $a_{e'} = 0$.

Suppose that $m \in V \setminus \{0, i, j\}$ and $l = i$ or $l = j$. Assume that $l = i$. Again, we know that $|V'| \geq 3$ and if $|V'| = 3$ then $V' = \{0, i, m\}$. By *ii)*, there exists $k \in V \setminus \{0, m, i, j\}$ such that $\beta_{km} = \beta_{ki} = \beta_{k0} = 0$. Let $x' = x + \sum_{e'' \in E(V' \cup \{k\})}(1 - x_{e''})\chi_{e''}$ and $y' = y - y_{km}\gamma_{km} - y_{ki}\gamma_{ki} - y_{k0}\gamma_{k0}$. As both solutions (x', y') and $(x' - \chi_{e'}, y')$ are in $\mathcal{F}$ we can conclude that $a_{e'} = 0$. If $|V'| = 4$ and $V' = \{0, m, i, v\}$ for some $v \in V \setminus \{0, m, i, j\}$ or $|V'| \geq 5$ then both solutions (x', y) with $x' = x + \sum_{e'' \in E(V') \setminus \{e\}}(1 - x_{e''})\chi_{e''}$ and $(x' - \chi_{e'}, y)$ are in $\mathcal{F}$ and hence $a_{e'} = 0$. The only remaining case is $V' = \{0, m, i, j\}$. By *ii)*, there exists $k \in V \setminus \{0, m, i, j\}$ such that $\beta_{km} = \beta_{ki} = \beta_{k0} = \beta_{kj} = 0$. Let $x^{'} = x + \sum_{e'' \in E(V' \cup \{k\}) \setminus \{e\}} (1 - x_{e''})\chi_{e''}$ and $y^{'} = y - y_{km}\gamma_{km} - y_{ki}\gamma_{ki} - y_{kj}\gamma_{kj} - y_{k0}\gamma_{k0}$. As $(x^{'}, y^{'})$ and $(x^{'} - \chi_{e'}, y^{'})$ are both in $\mathcal{F}$, we have $a_{e'} = 0$. The case with $l = j$ is the same.

Suppose that $e^{'} = \{0, m\}$ with $m \in V \setminus \{0, i, j\}$. This time, we know that $|V^{'}| \geq 2$. If $|V^{'}| = 2$, then $V^{'} = \{0, m\}$. Condition *iii)* implies that there exist two distinct nodes $k_1, k_2 \in V \setminus \{0, m\}$ such that $\beta_{k_1 m} = \beta_{k_1 0} = \beta_{k_2 m} = \beta_{k_2 0} = 0$. Consider the solution $(x^{'}, y^{'})$ with $x^{'} = x + \sum_{e'' \in E(V' \cup \{k_1, k_2\}) \setminus \{e\}} (1 - x_{e''})\chi_{e''}$ and $y^{'} = y - y_{k_1 m}\gamma_{k_1 m} - y_{k_1 0}\gamma_{k_1 0} - y_{k_2 m}\gamma_{k_2 m} - y_{k_2 0}\gamma_{k_2 0}$. As $(x^{'}, y^{'})$ and $(x^{'} - \chi_{e'}, y^{'})$ are in $\mathcal{F}$ we can conclude that $a_{e'} = 0$. If $|V^{'}| = 3$ and $V^{'} = \{0, m, v\}$ for some $v \in V \setminus \{0, m, i, j\}$, then by *i)*, we know that there exists $k \in V \setminus \{0, m, v\}$ such that $\beta_{km} = \beta_{kv} = \beta_{k0} = 0$. Let $x^{'} = x + \sum_{e'' \in E(V' \cup \{k\})} (1 - x_{e''})\chi_{e''}$ and $y^{'} = y - y_{km}\gamma_{km} - y_{kv}\gamma_{kv} - y_{k0}\gamma_{k0}$. Then as $(x^{'}, y^{'})$ and $(x^{'} - \chi_{e'}, y^{'})$ are both in $\mathcal{F}$, we have $a_{e'} = 0$. If $|V^{'}| = 3$ and $V^{'} = \{0, m, v\}$ for some $v \in \{i, j\}$, say without loss of generality that $v = i$, then by *ii)*, we know that there exists $k \in V \setminus \{0, m, i, j\}$ such that $\beta_{km} = \beta_{ki} = \beta_{k0} = 0$. Let $x^{'} = x + \sum_{e'' \in E(V' \cup \{k\})} (1 - x_{e''})\chi_{e''}$ and $y^{'} = y - y_{km}\gamma_{km} - y_{ki}\gamma_{ki} - y_{k0}\gamma_{k0}$. As both solutions $(x^{'}, y^{'})$ and $(x^{'} - \chi_{e'}, y^{'})$ are in $\mathcal{F}$, we can conclude that $a_{e'} = 0$. If $|V^{'}| \geq 4$, then $(x^{'}, y)$ with $x^{'} = x + \sum_{e'' \in E(V') \setminus \{e\}} (1 - x_{e''})\chi_{e''}$ and $(x^{'} - \chi_{e'}, y)$ are in $\mathcal{F}$. So $a_{e'} = 0$.

Suppose that $e^{'} = \{0, i\}$. If $|V^{'}| = 2$, then $V^{'} = \{0, i\}$. By *iv)*, there exist $k_1, k_2 \in V \setminus \{0, i, j\}$ such that $\beta_{k_1 i} = \beta_{k_1 0} = \beta_{k_2 i} = \beta_{k_2 0} = 0$. Both solutions $(x^{'}, y^{'})$ with $x^{'} = x + \sum_{e'' \in E(V' \cup \{k_1, k_2\})} (1 - x_{e''})\chi_{e''}$ and $y^{'} = y - y_{k_1 i}\gamma_{k_1 i} - y_{k_1 0}\gamma_{k_1 0} - y_{k_2 i}\gamma_{k_2 i} - y_{k_2 0}\gamma_{k_2 0}$ and $(x^{'} - \chi_{e'}, y^{'})$ are in $\mathcal{F}$. Thus we have $a_{e'} = 0$. If $|V^{'}| = 3$ and $V^{'} = \{0, i, v\}$ for some $v \neq j$ then by *ii)*, there exists $k \in V \setminus \{0, i, v, j\}$ such that $\beta_{ki} = \beta_{kv} = \beta_{k0} = 0$. Let $x^{'} = x + \sum_{e'' \in E(V' \cup \{k\})} (1 - x_{e''})\chi_{e''}$ and $y^{'} = y - y_{ki}\gamma_{km} - y_{kv}\gamma_{kv} - y_{k0}\gamma_{k0}$. Then both solutions $(x^{'}, y^{'})$ and $(x^{'} - \chi_{e'}, y^{'})$ are in $\mathcal{F}$. So $a_{e'} = 0$. If $|V^{'}| = 3$ and $V^{'} = \{0, i, j\}$, then by *iv)*, there exist $k_1, k_2 \in V \setminus \{0, i, j\}$ such that $\beta_{k_1 i} = \beta_{k_1 0} = \beta_{k_1 j} = \beta_{k_2 i} = \beta_{k_2 0} = \beta_{k_2 j} = 0$. Let $x^{'} = x + \sum_{e'' \in E(V' \cup \{k_1, k_2\}) \setminus \{e\}} (1 - x_{e''})\chi_{e''}$ and $y^{'} = y - y_{k_1 i}\gamma_{k_1 i} - y_{k_1 j}\gamma_{k_1 j} - y_{k_1 0}\gamma_{k_1 0} - y_{k_2 i}\gamma_{k_2 i} - y_{k_2 j}\gamma_{k_2 j} - y_{k_2 0}\gamma_{k_2 0}$. As the solutions $(x^{'}, y^{'})$ and $(x^{'} - \chi_{e'}, y^{'})$ are in $\mathcal{F}$, $a_{e'} = 0$. If $|V^{'}| = 4$ and $V^{'} = \{0, i, j, v\}$, then by *ii)*, there exists $k \in V \setminus \{0, i, v, j\}$ such that $\beta_{ki} = \beta_{kv} = \beta_{kj} = \beta_{k0} = 0$. Let $x^{'} = x + \sum_{e'' \in E(V' \cup \{k\}) \setminus \{e\}} (1 - x_{e''})\chi_{e''}$

and $y^{'} = y - y_{ki}\gamma_{ki} - y_{kv}\gamma_{kv} - y_{kj}\gamma_{kj} - y_{k0}\gamma_{k0}$. Both solutions $(x^{'}, y^{'})$ and $(x^{'} - \chi_{e^{'}}, y^{'})$ are in $\mathcal{F}$. So $a_{e^{'}} = 0$. If $|V^{'}| = 4$ and $j \notin V^{'}$ or if $|V^{'}| \geq 5$, then solutions $(x^{'}, y)$ with $x^{'} = x + \sum_{e^{''} \in E(V^{'}) \setminus \{e\}} (1 - x_{e^{''}})\chi_{e^{''}}$ and $(x^{'} - \chi_{e^{'}}, y)$ are in $\mathcal{F}$. Thus $a_{e^{'}} = 0$.

We proved that $ax + by = b_0$ is equal to $a_e x_e + by = b_0$.

Let $(m, l) \in A$ with $\beta_{ml} = 0$ and $(x, y) \in \mathcal{F}_{\mathcal{S}}$ be such that $y_{ml} = 1$. Let $V^{'} = \{v \in V : \sum_{k \in V \setminus \{v\}} y_{vk} = 0\} \cup \{0\}$. If $|V^{'}| = 1$, then $V^{'} = \{0\}$ and $l = 0$. If $m \neq i, j$ then by *iii)*, there exist two distinct nodes $k_1, k_2 \in V \setminus (\{0, m\})$ such that $|\{k_1, k_2\} \cap \{i, j\}| \leq 1$ and $\beta_{k_1 0} = \beta_{k_2 0} = 0$. If $m = i$ or $m = j$, then by *iv)*, there exist two distinct nodes $k_1, k_2 \in V \setminus (\{0, i, j\})$ such that $\beta_{k_1 0} = \beta_{k_2 0} = 0$. In both cases, the solution $(x^{'}, y^{'})$ where $x^{'} = x + \sum_{e^{'} \in E(\{0, k_1, k_2\})} (1 - x_{e^{'}})\chi_{e^{'}}$ and $y^{'} = y - y_{k_1 0}\gamma_{k_1 0} - y_{k_2 0}\gamma_{k_2 0}$ is in $\mathcal{F}$. Also the solution $(x^{'} + \sum_{e^{'} \in E(\{0, k_1, k_2, m\})} \chi_{e^{'}}, y^{'} - \gamma_{m0})$ is in $\mathcal{F}$, and hence we have $b_{m0} = 0$. If $|V^{'}| = 2$, then suppose $V^{'} = \{0, v\}$ for some $v \in V \setminus \{0, m\}$ such that $l \in V^{'}$. If $v \neq i, j$, then by *i)*, there exists a node $k \in V \setminus \{0, m, v\}$ such that $\beta_{k0} = \beta_{kv} = 0$. If $v \in \{i, j\}$, then by *ii)*, there exists a node $k \in V \setminus \{0, m, i, j\}$ such that $\beta_{k0} = \beta_{kv} = 0$. The solution $(x^{'}, y^{'})$ where $x^{'} = x + \sum_{e^{'} \in E(\{0, k, v\})} (1 - x_{e^{'}})\chi_{e^{'}}$ and $y^{'} = y - y_{k0}\gamma_{k0} - y_{kv}\gamma_{kv}$ is in $\mathcal{F}$. As the solution $(x + \sum_{e^{'} \in E(\{0, k, v, m\}) \setminus \{e\}} \chi_{e^{'}}, y^{'} - \gamma_{ml})$ is also in $\mathcal{F}$, we have $b_{ml} = 0$. If $|V^{'}| = 3$ and $V^{'} = \{0, i, j\}$, then by *ii)*, there exists a node $k \in V \setminus \{0, m, i, j\}$ such that $\beta_{k0} = \beta_{ki} = \beta_{kj} = 0$. Let $x^{'} = x + \sum_{e^{'} \in E(\{0, i, j, k\}) \setminus \{e\}} (1 - x_{e^{'}})\chi_{e^{'}}$ and $y^{'} = y - y_{k0}\gamma_{k0} - y_{ki}\gamma_{ki} - y_{kj}\gamma_{kj}$. Both solutions $(x^{'}, y^{'})$ and $(x + \sum_{e^{'} \in E(\{0, i, j, k, m\}) \setminus \{e\}} \chi_{e^{'}}, y^{'} - \gamma_{ml})$ are in $\mathcal{F}$, so we have $b_{ml} = 0$. Finally, if $|V^{'}| = 3$ and $|\{i, j\} \cap V^{'}| \leq 1$ or $|V^{'}| \geq 4$, then $(x^{'}, y)$ and $(x^{'} + \sum_{e^{'} \in E(V^{'} \cup \{m\}) \setminus \{e\}} \chi_{e^{'}}, y - \gamma_{ml})$ are both in $\mathcal{F}$, where $x^{'} = x + \sum_{e^{'} \in E(V^{'}) \setminus \{e\}} (1 - x_{e^{'}})\chi_{e^{'}}$. So we can conclude that $b_{ml} = 0$. Therefore, $b_{ml} = 0$ for all $(m, l) \in A$ with $\beta_{ml} = 0$.

Now assume that there exists $(x, y) \in \mathcal{F}_{\mathcal{S}}$ such that $a_e x_e + by \neq b_0$. Let $V^{'} = \{v \in V : \sum_{k \in V \setminus \{v\}} y_{vk} = 0\} \cup \{0\}$. Unless $|V^{'}| = 2$ or $V^{'} = \{0, i, j\}$ and $x_e = 0$, the solution $(x^{'}, y)$ where $x^{'} = x + \sum_{e^{'} \in E(V^{'}) \setminus \{e\}} (1 - x_{e^{'}})\chi_{e^{'}}$ is in $\mathcal{F}$ and $a_e x^{'}_e + by \neq b_0$. Thus, either $|V^{'}| = 2$ or $V^{'} = \{0, i, j\}$ and $x_e = 0$. First suppose that $|V^{'}| = 2$ and that $V^{'} = \{0, m\}$. If $m \neq i, j$, then by *iii)*, there exists a node $k \in V \setminus \{0, m\}$ such that $\beta_{km} = \beta_{k0} = 0$. If $m \in \{i, j\}$, then by *iv)*, there exists a node $k \in V \setminus \{0, i, j\}$ such that $\beta_{km} = \beta_{k0} = 0$. In both cases, the solution $(x^{'}, y^{'})$

where $x^{'} = x+\sum_{e'\in E(\{0,m,k\})}(1-x_{e'})\chi_{e'}$ and $y^{'} = y-y_{k0}\gamma_{k0}-y_{km}\gamma_{km}$ is in $\mathcal{F}$ and $a_e x^{'}_e + by^{'} \neq b_0$. Now suppose that $V^{'} = \{0,i,j\}$ and $x_e = 0$. By *iv)*, there exists a node $k \in V \setminus \{0,i,j\}$ such that $\beta_{ki} = \beta_{kj} = \beta_{k0} = 0$. Now the solution $(x^{'}, y^{'})$ where $x^{'} = x+\sum_{e'\in E(\{0,i,j,k\})\setminus\{e\}}(1-x_{e'})\chi_{e'}$ and $y^{'} = y-y_{k0}\gamma_{k0}-y_{ki}\gamma_{ki}-y_{kj}\gamma_{kj}$ is in $\mathcal{F}$ and $a_e x^{'}_e+by^{'} \neq b_0$, a contradiction. So we can conclude that all solutions $(x,y) \in \mathcal{F}_{\mathcal{S}}$ satisfy $a_e x_e + by = b_0$ and hence $a_e x_e + by = b_0$ is a multiple of $\alpha_e x_e + \beta y = \beta_0$. $\square$

Theorem 3.7 *Let $e = \{0,i\} \in E$. Suppose that the inequality $\alpha_e x_e + \beta y \leq \beta_0$ is facet defining for $\mathcal{P}_{\mathcal{S}}$. If*

i. for all $m \in V \setminus \{0,i\}$ and $l \in V \setminus \{0,i,m\}$, there exists a node $k \in V \setminus \{0,i,m,l\}$ such that $\beta_{ki} = \beta_{km} = \beta_{kl} = \beta_{k0} = 0$,

ii. for $m \in V \setminus \{0,i\}$, there exist two distinct nodes $k_1, k_2 \in V \setminus \{0,i,m\}$ such that $\beta_{k_1 m} = \beta_{k_1 i} = \beta_{k_1 0} = \beta_{k_2 m} = \beta_{k_2 i} = \beta_{k_2 0} = 0$,

are all satisfied, then the inequality $\alpha_e x_e + \beta y \leq \beta_0$ also defines a facet of $\mathcal{P}$.

Proof Let $e = \{0,i\} \in E$. Suppose that the inequality $\alpha_e x_e + \beta y \leq \beta_0$ is facet defining for $\mathcal{P}_{\mathcal{S}}$ and that conditions *(i)* and *(ii)* are satisfied. Define $\mathcal{F}_{\mathcal{S}} = \{(x,y) \in P_S : \alpha_e x_e + \beta y = \beta_0\}$ and $\mathcal{F} = \{(x,y) \in P : \alpha_e x_e + \beta y = \beta_0\}$. Suppose that all solutions $(x,y) \in \mathcal{F}$ also satisfy $ax+by = b_0$. Let $e^{'} = \{m,l\} \in E \setminus \{e\}$. There exists a solution $(x,y) \in \mathcal{F}_{\mathcal{S}}$ such that $x_{e^{'}} = 1$. Let $V^{'} = \{v \in V : \sum_{k\in V\setminus\{v\}} y_{vk} = 0\} \cup \{0\}$.

Suppose that $m \in V \setminus \{0,i\}$ and $l \in V \setminus \{0,i,m\}$. As $x_{e^{'}} = 1$, we know that $|V^{'}| \geq 3$. If $|V^{'}| = 3$ then $V^{'} = \{0,m,l\}$. By *(i)*, there exists $k \in V\setminus\{0,m,l\}$ such that $\beta_{km} = \beta_{kl} = \beta_{k0} = 0$. The solution $(x^{'}, y^{'})$ with $x^{'} = x+\sum_{e''\in E(V'\cup\{k\})\setminus\{e\}}(1-x^{e^{''}})\chi_{e''}$ and $y^{'} = y-y_{km}\gamma_{km}-y_{kl}\gamma_{kl}-y_{k0}\gamma_{k0}$ is in $\mathcal{F}$. Also the solution $(x^{'}-\chi_{e^{'}}, y^{'})$ is in $\mathcal{F}$ and hence $a_{e^{'}} = 0$. If $|V^{'}| \geq 4$ then both solutions $(x^{'}, y)$ with $x^{'} = x+\sum_{e''\in E(V')\setminus\{e\}}(1-x^{e^{''}})\chi_{e''}$ and $(x^{'}-\chi_{e^{'}}, y)$ are in $\mathcal{F}$ implying that $a_{e^{'}} = 0$.

Suppose that that $m \in V \setminus \{0,i\}$ and $l = i$. Then $x_{e^{'}} = 1$ implies that $|V^{'}| \geq 3$. If $|V^{'}| = 3$ then $V^{'} = \{0,i,m\}$. By *(ii)*, there exist two distinct nodes $k_1, k_2 \in V \setminus \{0, i, m\}$ such that $\beta_{k_1m} = \beta_{k_1i} = \beta_{k_10} = \beta_{k_2m} = \beta_{k_2i} = \beta_{k_20} = 0$. Both solutions (x', y') with $x' = x + \sum_{e'' \in E(V' \cup \{k_1,k_2\}) \setminus \{e\}} (1 - x^{e''})\chi_{e''}$ and $y' = y - y_{k1m}\gamma_{k1m} - y_{k1i}\gamma_{k1i} - y_{k10}\gamma_{k10} - y_{k2m}\gamma_{k2m} - y_{k2i}\gamma_{k2i} - y_{k20}\gamma_{k20}$ and $(x' - \chi_{e'}, y')$ are in $\mathcal{F}$. So $a_{e'} = 0$. If $|V'| = 4$, then $V' = \{0, m, i, v\}$ for some $v \in V \setminus \{0, i, m\}$. By *(i)*, there exists $k \in V \setminus \{0, i, m, v\}$ such that $\beta_{ki} = \beta_{km} = \beta_{kv} = \beta_{k0} = 0$. Then as (x', y') with $x' = x + \sum_{e'' \in E(V' \cup \{k\}) \setminus \{e\}} (1 - x^{e''})\chi_{e''}$ and $y' = y - y_{km}\gamma_{km} - y_{ki}\gamma_{ki} - y_{kv}\gamma_{kv} - y_{k0}\gamma_{k0}$ and $(x' - \chi_{e'}, y')$ are in $\mathcal{F}$, we have $a_{e'} = 0$. If $|V'| \geq 5$ then both solutions (x', y) with $x' = x + \sum_{e'' \in E(V') \setminus \{e\}} (1 - x^{e''})\chi_{e''}$ and $(x' - \chi_{e'}, y)$ are in $\mathcal{F}$ and hence $a_{e'} = 0$.

Suppose that that $m \in V \setminus \{0, i\}$ and $l = 0$. As $x_{e'} = 1$, we have $|V'| \geq 2$. If $|V'| = 2$, then $V' = \{0, m\}$. By *(ii)*, there exist two distinct nodes $k_1, k_2 \in V \setminus \{0, i, m\}$ such that $\beta_{k_1m} = \beta_{k_10} = \beta_{k_2m} = \beta_{k_20} = 0$. Both solutions (x', y') with $x' = x + \sum_{e'' \in E(V' \cup \{k_1,k_2\})} (1 - x^{e''})\chi_{e''}$ and $y' = y - y_{k_1m}\gamma_{k_1m} - y_{k_10}\gamma_{k_10} - y_{k_2m}\gamma_{k_2m} - y_{k_20}\gamma_{k_20}$ and $(x' - \chi_{e'}, y')$ are in $\mathcal{F}$. Thus $a_{e'} = 0$. If $|V'| = 3$ and $V' = \{0, m, v\}$ for some $v \neq i$ then by *(i)*, we know that there exists $k \in V \setminus \{0, i, m, v\}$ such that $\beta_{km} = \beta_{kv} = \beta_{k0} = 0$. Then as (x', y') with $x' = x + \sum_{e'' \in E(V' \cup \{k\})} (1 - x^{e''})\chi_{e''}$ and $y' = y - y_{km}\gamma_{km} - y_{kv}\gamma_{kv} - y_{k0}\gamma_{k0}$ and $(x' - \chi_{e'}, y')$ are in $\mathcal{F}$, we have $a_{e'} = 0$. If $|V'| = 3$ and $V' = \{0, i, m\}$, then by *(ii)*, we know that there exist two distinct nodes $k_1, k_2 \in V \setminus \{0, i, m\}$ such that $\beta_{k_1m} = \beta_{k_1i} = \beta_{k_10} = \beta_{k_2m} = \beta_{k_2i} = \beta_{k_20} = 0$. Again as (x', y') with $x' = x + \sum_{e'' \in E(V' \cup \{k_1,k_2\}) \setminus \{e\}} (1 - x^{e''})\chi_{e''}$ and $y' = y - y_{k_1i}\gamma_{k_1i} - y_{k_1m}\gamma_{k_1m} - y_{k_10}\gamma_{k_10} - y_{k_2i}\gamma_{k_2i} - y_{k_2m}\gamma_{k_2m} - y_{k_20}\gamma_{k_20}$ and $(x' - \chi_{e'}, y')$ are in $\mathcal{F}$, we have $a_{e'} = 0$. If $|V'| = 4$ and $V' = \{0, i, m, v\}$, then by *(i)*, we know that there exists $k \in V \setminus \{0, i, m, v\}$ such that $\beta_{ki} = \beta_{km} = \beta_{kv} = \beta_{k0} = 0$. Now as (x', y') with $x' = x + \sum_{e'' \in E(V' \cup \{k\}) \setminus \{e\}} (1 - x^{e''})\chi_{e''}$ and $y' = y - y_{ki}\gamma_{ki} - y_{km}\gamma_{km} - y_{kv}\gamma_{kv} - y_{k0}\gamma_{k0}$ and $(x' - \chi_{e'}, y')$ are in $\mathcal{F}$, we have $a_{e'} = 0$. If $|V'| = 4$ and $i \notin V'$ or $|V'| \geq 5$. then (x', y) with $x' = x + \sum_{e'' \in E(V') \setminus \{e\}} (1 - x^{e''})\chi_{e''}$ and $(x' - \chi_{e'}, y)$ are in $\mathcal{F}$. So $a_{e'} = 0$.

So $a_{e'} = 0$ for all $e' \in E \setminus \{e\}$.

Let $(m, l) \in A$ with $\beta_{ml} = 0$ and $(x, y) \in \mathcal{F}_{\mathcal{S}}$ be such that $y_{ml} = 1$. Let $V' = \{v \in V : \sum_{k \in V \setminus \{v\}} y_{vk} = 0\} \cup \{0\}$. If $|V'| = 1$, then $V' = \{0\}$ and $l = 0$. By *(ii)*, there exist two distinct nodes $k_1, k_2 \in V \setminus (\{0, i, m\})$ such that $\beta_{k_10} = \beta_{k_20} = 0$

(if $m = i$, then for some other node $v \in V \setminus \{0, i\}$, *(ii)* implies that there exist two distinct nodes $k_1, k_2 \in V\setminus(\{0, i, v\})$ such that $\beta_{k_10} = \beta_{k_20} = 0$). Then the solution $(x^{'}, y^{'})$ where $x^{'} = x+\sum_{e^{'}\in E(\{0,k_1,k_2\})}(1-x^{e^{'}})\chi_{e^{'}}$ and $y^{'} = y-y_{k_10}\gamma_{k_10}-y_{k_20}\gamma_{k_20}$ is in $\mathcal{F}$. Also the solution $(x^{'}+\sum_{e^{'}\in E(\{0,k_1,k_2,m\}))\setminus\{e\}}\chi_{e^{'}}, y^{'}-\gamma_{ml})$ is in $\mathcal{F}$ and hence we have $b_{ml} = 0$. If $|V^{'}| = 2$, then suppose $V^{'} = \{0, v\}$ for some $v \in V \setminus \{0, m\}$ such that $l \in V^{'}$. If $v \neq i$, then if $m \neq i$ by *(i)* and if $m = i$ by *(ii)*, there exists a node $k \in V \setminus \{0, i, m, v\}$ such that $\beta_{kv} = \beta_{k0} = 0$. Both solutions $(x^{'}, y^{'})$ where $x^{'} = x + \sum_{e^{'}\in E(\{0,k,v\})}(1 - x^{e^{'}})\chi_{e^{'}}$ and $y^{'} = y - y_{k0}\gamma_{k0} - y_{kv}\gamma_{kv}$ and $(x^{'} + \sum_{e^{'}\in E(\{0,k,v,m\})\setminus\{e\}}\chi_{e^{'}}, y^{'} - \gamma_{ml})$ is also in $\mathcal{F}$, we have $b_{ml} = 0$. If $v = i$, then by *(ii)*, there exist two distinct nodes $k_1, k_2 \in V \setminus (\{0, i, m\})$ such that $\beta_{k_10} = \beta_{k_1i} = \beta_{k_20} = \beta_{k_2i} = 0$. Then the solutions $(x^{'}, y^{'})$ where $x^{'} = x + \sum_{e^{'}\in E(\{0,i,k_1,k_2\})\setminus\{e\}}(1 - x^{e^{'}})\chi_{e^{'}}$ and $y^{'} = y - y_{k_10}\gamma_{k_10} - y_{k_1i}\gamma_{k_1i} - y_{k_20}\gamma_{k_20} - y_{k_2i}\gamma_{k_2i}$ and $(x^{'} + \sum_{e^{'}\in E(\{0,i,m,k_1,k_2\})\setminus\{e\}}\chi_{e^{'}}, y^{'} - \gamma_{ml})$ are in $\mathcal{F}$ and hence we have $b_{ml} = 0$. If $|V^{'}| = 3$ and $V^{'} = \{0, i, v\}$, then by *(i)*, there exists a node $k \in V \setminus \{0, m, i, v\}$ such that $\beta_{k0} = \beta_{ki} = \beta_{kv} = 0$. Both solutions $(x^{'}, y^{'})$ where $x^{'} = x + \sum_{e^{'}\in E(\{0,i,v,k\})\setminus\{e\}}(1 - x^{e^{'}})\chi_{e^{'}}$ and $y^{'} = y - y_{k0}\gamma_{k0} - y_{ki}\gamma_{ki} - y_{kv}\gamma_{kv}$ and $(x^{'} + \sum_{e^{'}\in E(\{0,i,v,k,m\})\setminus\{e\}}\chi_{e^{'}}, y^{'} - \gamma_{ml})$ are in $\mathcal{F}$, we have $b_{ml} = 0$. If $|V^{'}| = 3$ and $i \notin V^{'}$ or $|V^{'}| \geq 4$, then $(x^{'}, y)$ where $x^{'} = x + \sum_{e^{'}\in E(V^{'})\setminus\{e\}}(1 - x^{e^{'}})\chi_{e^{'}}$ and $(x^{'} + \sum_{e^{'}\in E(V^{'}\cup\{m\})\setminus\{e\}}\chi_{e^{'}}, y - \gamma_{ml})$ are in $\mathcal{F}$, we can conclude that $b_{ml} = 0$. Therefore, $b_{ml} = 0$ for all $(m, l) \in A$ with $\beta_{ml} = 0$.

Now assume that there exists $(x, y) \in \mathcal{F_S}$ such that $a_ex_e + by \neq b_0$. Let $V^{'} = \{v \in V : \sum_{k\in V\setminus\{v\}} y_{vk} = 0\} \cup \{0\}$. Unless $|V^{'}| = 2$ or $V^{'} = \{0, i, v\}$ for some $v \in V \setminus \{0, i\}$ and $x_e = 0$, the solution $(x^{'}, y)$ where $x^{'} = x + \sum_{e^{'}\in E(V^{'})\setminus\{e\}}(1 - x^{e^{'}})\chi_{e^{'}}$ is in $\mathcal{F}$ and $a_ex^{'}_e + by \neq b_0$. So either $|V^{'}| = 2$ or $V^{'} = \{0, i, v\}$ for some $v \in V \setminus \{0, i\}$ and $x_e = 0$. First suppose that $|V^{'}| = 2$ and that $V^{'} = \{0, v\}$. If $v \neq i$, then by *(i)*, there exists a node $k \in V \setminus \{0, i, v\}$ such that $\beta_{kv} = \beta_{k0} = 0$. Then the solution $(x^{'}, y^{'})$ where $x^{'} = x + \sum_{e^{'}\in E(\{0,v,k\})}(1 - x^{e^{'}})\chi_{e^{'}}$ and $y^{'} = y - y_{k0}\gamma_{k0} - y_{kv}\gamma_{kv}$ is in $\mathcal{F}$ and $a_ex^{'}_e + by^{'} \neq b_0$. If $v = i$, then by *(ii)*,there exist two distinct nodes $k_1, k_2 \in V \setminus \{0, i\}$ such that $\beta_{k1i} = \beta_{k10} = \beta_{k2i} = \beta_{k20} = 0$. Then the solution $(x^{'}, y^{'})$ where $x^{'} = x+\sum_{e^{'}\in E(\{0,i,k_1,k_2\})\setminus\{e\}}(1-x^{e^{'}})\chi_{e^{'}}$ and $y^{'} = y-y_{k_10}\gamma_{k_10}-y_{k_1i}\gamma_{k_1i}-y_{k_20}\gamma_{k_20}-y_{k_2i}\gamma_{k_2i}$ is in $\mathcal{F}$ and $a_ex^{'}_e+by^{'} \neq b_0$. Now suppose that $V^{'} = \{0, i, v\}$ for some $v \in V \setminus \{0, i\}$ and $x_e = 0$. Then by *(ii)*, there exists a node $k \in V \setminus \{0, i, v\}$ such that $\beta_{ki} = \beta_{kv} = \beta_{k0} = 0$. Then the solution $(x^{'}, y^{'})$ where $x^{'} = x + \sum_{e' \in E(\{0,v,i,k\}) \setminus \{e\}} (1 - x^{e^{'}}) \chi_{e'}$ and $y^{'} = y - y_{k0}\gamma_{k0} - y_{kv}\gamma_{kv} - y_{ki}\gamma_{ki}$ is in $\mathcal{F}$ and $a_e x_e^{'} + by^{'} \neq b_0$. Thus all solutions $(x, y) \in \mathcal{F}_S$ satisfy $a_e x_e + by \neq b_0$ and so $a_e x_e + by \neq b_0$ is a multiple of $\alpha_e x_e + \beta y = \beta_0$. $\square$

We can use the information about the stable set and the conflict graph. Note that a conflict graph is a graph that includes an edge between two nodes if these nodes cannot be found in a solution simultaneously. Consider the conflict graph associated with the set $X_S = \{(x, y) \in R^{|E|+|A|} : (x, y)$ satisfies (3.10), (3.11), (3.15), and (3.16)$\}$. This conflict graph can be used to obtain some valid inequalities for $\mathcal{P}$.

Let i, j, k be distinct nodes in $V \setminus \{0\}$. Then inequality $y_{ij} + y_{jk} + y_{ki} \leq 1$ is valid for $\mathcal{P}$. Clearly, if i is assigned to j, then i is a terminal and j is a concentrator. This implies that k cannot be assigned to i and j cannot be assigned to any node. The cases $y_{jk} =$ and $y_{ki} = 1$ are similar. This inequality is known as the *triangle inequality.*

It is known that a clique inequality is facet defining for the stable set polytope if and only if the underlying clique is maximal [4]. Now, we investigate the maximal cliques in the conflict graph associated with X_S.

Theorem 3.8 *The only facet defining clique inequalities for $\mathcal{P}_S$ are constraints (3.10) and (3.11), and inequalities $y_{ij} + y_{jk} + y_{ki} \leq 1$ for distinct nodes $i, j, k \in V \setminus \{0\}$.*

Proof Consider a maximal clique in the conflict graph associated with X_S. Observe that this clique can contain at most one x_e variable. First suppose that x_{ij} is in the clique for some $\{i, j\} \in E$ such that $i \neq 0$ and $j \neq 0$. Then the neighbors of x_{ij} are nodes of the form y_{il} for $l \in V \setminus \{i\}$ and y_{jk} for $k \in V \setminus \{j\}$. Assume without loss of generality that the clique contains a node y_{il} for some $l \in V \setminus \{i\}$. Then there exists a node of the form y_{jk} for some $k \in V \setminus \{j\}$ in the clique only if $l = j$. If $l = j$, then the clique contains all nodes of the form y_{jk} for $k \in V \setminus \{j\}$

and no other node. The corresponding clique inequality is (3.10). Now suppose that the clique does not contain any node of the form y_{jk} for $k \in V \setminus \{j\}$. Then it contains all nodes y_{il} for $l \in V \setminus \{i\}$ and no other node. However, such a clique cannot be maximal as it can be enlarged by adding the node y_{ji} and hence, we arrive at a contradiction.

Now suppose that x_{0i} is in the clique for some $i \in V \setminus \{0\}$. The neighbors of x_{0i} are nodes of the form y_{ik} for $k \in V \setminus \{i\}$. The node x_{0i} together with the nodes y_{ik} for all $k \in V \setminus \{i\}$ form a maximal clique and the corresponding clique inequality is (3.11).

The remaining maximal cliques do not include any node of the form x_e for $e \in E$. Suppose that we have such a maximal clique which includes a node y_{ij} for some $(i, j) \in A$. If the clique also includes a node of the form y_{il} for $l \in V \setminus \{i, j\}$, then it can only include the other nodes y_{ik} for $k \in V \setminus \{i, j, l\}$ and a node y_{mi} for some $m \in V \setminus \{0, i\}$. Such a clique cannot be maximal as it can be extended by adding the node x_{mi}. The neighbors of y_{ij} other than those of the form x_e for $e \in E$ and y_{il} for $l \in V \setminus \{i, j\}$ are the nodes y_{jk} for some $k \in V \setminus \{j\}$ if $j \neq 0$. If $j = 0$, then there is no such neighbor. Suppose that $j \neq 0$ and that the clique contains y_{ij} and y_{jk} for some $k \in V \setminus \{j\}$. If the clique contains another node y_{jl} for $l \in V \setminus \{j, k\}$, then it can only include the other nodes y_{jm} for $m \in V \setminus \{j, k, l\}$ and can be extended by adding the node x_{ij}. Therefore, if a maximal clique includes nodes y_{ij} and y_{jk} and does not include any node of the form x_e, y_{il} for $l \in V \setminus \{i, j\}$, and y_{jl} for $l \in V \setminus \{j, k\}$, then it should contain the node y_{ki}. The associated clique inequality is $y_{ij} + y_{jk} + y_{ki} \leq 1$. $\square$

As consequences of Theorems 3.6, 3.7, and 3.8 we have the following:

Corollary 3.1 *Let $(i, j) \in A$ with $j \neq 0$. Then inequality (3.10) is facet defining for $\mathcal{P}$.*

Corollary 3.2 *Let $i \in V \setminus \{0\}$. Then inequality (3.11) is facet defining for $\mathcal{P}$.*

Corollary 3.3 *Let i, j, and k be distinct nodes in $V \setminus \{0\}$. Then inequality $y_{ij} + y_{jk} + y_{ki} \leq 1$ is facet defining for $\mathcal{P}$.*

With these corollaries the polyhedral analysis of the constraints of the formulation is completed. Corollaries 3.1 and 3.2 together with Theorem 3.5 show that all the constraints of the model are facet defining for the polytope $\mathcal{P}$. We continue with the analysis of additional classes of valid inequalities.

3.2.5 Extended F-partition inequalities

F-partition inequalities form an important class of valid inequalities for the 2-edge connected subgraph problem. These inequalities are shown to be very effective for solving large instances of the 2-edge connected subgraph problem (see [27, 34]). We observed that they can be extended to the 2ECSSP polytope $\mathcal{P}$.

An example of a fractional solution that can be cut off by the extension of F-partition inequalities may be useful. Consider the solution $(\overline{x}, \overline{y})$ depicted in Figure 3.2. Let $V = \{0, \ldots, 9\}$. We omit the edges and arcs if the value of corresponding variables are 0. The positive values in $(\overline{x}, \overline{y})$ are either 0.5 or 1. The backbone edges with value 1 are represented by bold lines and those with value 0.5 are represented by dashed lines. The assignments are as follows: $\overline{y}_{ii} = 1$ for $i \in V \setminus \{3, 4\}$ (these nodes are represented by rectangles) and $\overline{y}_{33} = \overline{y}_{35} = \overline{y}_{44} = \overline{y}_{46} = 0.5$ (these nodes are represented by triangles and the assignments of 3 to 5 and 4 to 6 are represented by dashed lines with arrows). The solution $(\overline{x}, \overline{y})$ satisfies all the clique (3.10), (3.11), and cut (3.12) inequalities and is an extreme point of the linear relaxation of 2ECSSP.

Consider a partition of V into $V_0, \ldots, V_5$ such that $V_0 = \{0, 8, 9\}$, $V_1 = \{1\}$, $V_2 = \{2\}$, $V_3 = \{3, 5\}$, $V_4 = \{4, 6\}$, and $V_5 = \{7\}$. Let $F = \{\{0, 7\}, \{2, 8\}, \{5, 9\}\}$. Each set in the partition has at least one node at which a concentrator is installed. In order to have 2-edge connectedness among the sets of the partition, at least 4 edges from the set $\delta(V_0, \ldots, V_5) \setminus F$ must be used and this implies $x(\delta(V_0, \ldots, V_5) \setminus F) \geq 4$. Notice that as the root node is in V_0, there is a concentrator installed

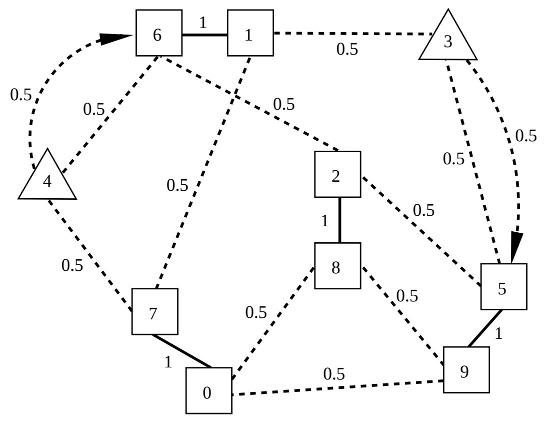

Figure 3.2: A fractional solution cut off by an F-partition inequality.

in the set V_0 in any fractional solution. This is not necessarily true for the remaining sets of the partition. For instance if node 1 is assigned to another node, then there is no concentrator installed in set V_1 and we need 3 edges from the set $\delta(V_0, \ldots, V_5) \setminus F$ for 2-edge connectedness. So we need $x(\delta(V_0, \ldots, V_5) \setminus F) + \sum_{j \in V \setminus \{1\}} y_{1j} \geq 4$. We can repeat the same argument for the remaining sets of the partition. Here for sets that are not singletons, we can only use one node in the inequality. Suppose that we pick node 3 for set V_3 and node 4 for set V_4. We obtain the inequality

$$x(\delta(V_0, \ldots, V_5) \setminus F) + \sum_{j \in V \setminus \{1\}} y_{1j} + \sum_{j \in V \setminus \{2\}} y_{2j} + \sum_{j \in V \setminus \{3,5\}} y_{3j} + \sum_{j \in V \setminus \{4,6\}} y_{4j} + \sum_{j \in V \setminus \{7\}} y_{7j} \geq 4$$

which is a valid inequality. This inequality cuts off the fractional solution $(\overline{x}, \overline{y})$ since

$$\begin{aligned} &\overline{x}(\delta(V_0, \ldots, V_5) \setminus F) + \textstyle\sum_{j \in V \setminus \{1\}} \overline{y}_{1j} + \sum_{j \in V \setminus \{2\}} \overline{y}_{2j} + \sum_{j \in V \setminus \{3,5\}} \overline{y}_{3j} \\ &\quad + \textstyle\sum_{j \in V \setminus \{4,6\}} \overline{y}_{4j} + \sum_{j \in V \setminus \{7\}} \overline{y}_{7j} \\ &= \overline{x}_{13} + \overline{x}_{16} + \overline{x}_{17} + \overline{x}_{25} + \overline{x}_{26} + \overline{x}_{47} = 3.5 < 4. \end{aligned}$$

Now we can give the formal definition of the extended version of the F-partition inequalities.

Let $V_0, \ldots, V_p$ be a partition of V such that $V_l \neq \emptyset$, for $l = 0, \ldots, p$ and $0 \in V_0$. Let $i_l \in V_l$ be a fixed node for $l = 1, \ldots, p$ and $F \subseteq \delta(V_0)$ such that $|F| = 2k + 1$ for some $k \geq 0$ and integer. Let $\delta(V_0, \ldots, V_p)$ be the set of edges whose endpoints are in different sets of the partition. Consider the inequality

$$x(\delta(V_0, \ldots, V_p) \setminus F) + \sum_{l=1}^{p} \sum_{j \in V \setminus V_l} y_{i_l j} \geq p - k. \tag{3.18}$$

The following theorem shows the validity of inequality (3.18).

Theorem 3.9 *Inequality (3.18) is valid for $\mathcal{P}$.*

Proof The following inequalities are valid for $\mathcal{P}$:

$$\begin{aligned} x(\delta(V_l)) + 2 \sum_{j \in V \setminus V_l} y_{i_l j} &\geq 2 && l = 1, \ldots, p \\ -x_e &\geq -1 && \forall e \in F \\ x_e &\geq 0 && \forall e \in \delta(V_0) \setminus F. \end{aligned}$$

Adding up these inequalities and dividing the resulting inequality by 2 yields

$$x(\delta(V_0, \ldots, V_p) \setminus F) + \sum_{l=1}^{p} \sum_{j \in V \setminus V_l} y_{i_l j} \geq p - \frac{|F|}{2}$$

As $|F|$ is odd, rounding up the right hand side yields inequality (3.18). $\square$

Inequalities of type (3.18) will be called *extended F-partition inequalities.* Note that, if values of all assignment variables are zero, i.e., if all nodes are selected as hubs, then the extended F-partition inequalities are the same as the F-partition inequalities of 2-edge connected subgraph problem.

The extended F-partition inequalities define facets for $\mathcal{P}$ under some conditions. We provide some sufficient conditions for them to be facet defining. We will show a different method, namely the lifting method, in establishing the sufficient conditions under which the extended F-partition inequalities define facets. We need the following information in the lifting method.

For $\overline{A} \subseteq A$, let $X_{\overline{A}} = \{(x,y) \in X : y_a = 0 \ \forall a \in A \setminus \overline{A}\}$ and $\mathcal{P}_{\overline{A}} = conv(X_{\overline{A}})$. Suppose that $\alpha x + \beta y \geq \xi$ is a facet defining inequality for $\mathcal{P}_{\overline{A}}$. Let $a \in A \setminus \overline{A}$ and $\overline{A}' = \overline{A} \cup \{a\}$. Then the inequality

$$\alpha x + \beta y + b_a y_a \geq \xi$$

is facet defining for $\mathcal{P}_{\overline{A}'}$ where $b_a = \xi - \theta_a(\overline{A}')$ and $\theta_a(\overline{A}') = \min\{\alpha x + \beta y : (x,y) \in X_{\overline{A}'}$ and $y_a = 1\}$ [37].

Theorem 3.10 *Inequality (3.18) defines a facet for $\mathcal{P}$ if the following conditions are all satisfied*

a) $G(V_l)$ is 3-edge connected for $l = 0, \ldots, p$,

b) $|F \cap \delta(V_l)| \leq 1$ and $F \cap \delta(j) = \emptyset$ for $l = 1, \ldots, p$ and $j \in V_l \setminus \{i_l\}$,

c) $|F \cap \delta(j)| \leq 1$ for $j \in V_0 \setminus \{0\}$.

Proof For simplicity, we use Δ, L, and I to denote $\delta(V_0, \ldots, V_p) \setminus F$, $\{1, \ldots, p\}$, and $\{i_1, \ldots, i_p\}$, respectively. Without loss of generality we assume that $\delta(i_l) \cap F \neq \emptyset$ for $l = 1, \ldots, 2k$. Note that from Condition b), we have $2k + 1 \leq p$.

For $\overline{A} = \emptyset$, $\mathcal{P}_{\overline{A}}$ reduces to the 2-edge connected subgraph polytope. Since $G(V_l)$ is 3-edge connected for $l = 0, \ldots, p$ and $G = (V, E)$ is complete, from [34] it follows that $x(\Delta) \geq p - k$ is a facet defining inequality for $\mathcal{P}_{\overline{A}}$.

If $p > 2k+1$, we let $E_1 = \cup_{l=0}^{p} E(V_l) \cup \{i_1, i_{2k+1}\} \cup \{i_2, i_p\} \cup_{l=2k+1}^{p-1} \{i_l, i_{l+1}\} \cup_{l=2}^{k} \{i_{2l-1}, i_{2l}\} \cup F$ and $x = \sum_{e \in E_1} \chi_e$. Clearly, $(x, 0) \in X$. If $p = 2k + 1$, let $E_2 = \cup_{l=0}^{p} E(V_l) \cup_{l=2}^{k} \{i_{2l}, i_{2l+1}\} \cup \{i_1, i_2\} \cup \{i_2, i_3\} \cup F$. Then $(\sum_{e \in E_2} \chi_e, 0) \in X$. Here we give the proof for the case where $p > 2k + 1$. The proof for the other case is similar.

Let $A_1 = \{(u,v) \in A : u \in V \setminus I, v \in V\} \cup \{(u,v) \in A : u = i_l$ for some $l \in L, v \in V_l\}$. We first show that the lifting coefficients of the variables associated with the arcs of A_1 are zero. The proof is by induction. Let $(u,v) \in A_1$ be the first arc in the lifting sequence. Then

$$b_{uv} = p - k - \theta_{(u,v)}(\{(u,v)\}).$$

Inequality (3.18) implies that $\theta_{(u,v)}(\{(u,v)\}) \geq p-k$. Let $x_1 = x - \sum_{e\in\delta(u)} x_e\chi_e$. Suppose that $u \notin I$. Note that if $u \in V_0$ and there exists $f \in F$ with $u \in f$, then we can rearrange the partition subsets so that $f \in \delta(V_{2k+1})$ and hence $(x_1, \gamma_{uv}) \in X_{\{(u,v)\}}$. If $u \in I$, then without loss of generality, we may assume that $u = i_{2k+1}$. Therefore $(x_1 + \chi_{i_1 v} + \chi_{v i_{2k+2}}, \gamma_{uv}) \in X_{\{(u,v)\}}$. In both cases, $\theta_{(u,v)}(\{(u,v)\}) = p-k$ and thus $b_{uv} = 0$. In consequence $x(\Delta) \geq p-k$ is facet defining for $\mathcal{P}_{\{(u,v)\}}$. Now, let $\overline{A}_1 \subseteq A_1$ be the set of arcs for which the lifting has already been done and $(u,v) \in A_1 \setminus \overline{A}_1$. We assume that $b_a = 0$ for every $a \in \overline{A}_1$ and show that $b_{uv} = 0$. Here $b_{uv} = p-k-\theta_{(u,v)}(\overline{A}_1 \cup \{(u,v)\})$. Clearly, by inequality (3.18) $\theta_{(u,v)}(\overline{A}_1 \cup \{(u,v)\}) \geq p-k$. Using the same approach as above, we can similarly show that $\theta_{(u,v)}(\overline{A}_1 \cup \{(u,v)\}) = p-k$, and thus $b_{uv} = 0$. So $x(\Delta) \geq p-k$ is facet defining for $\mathcal{P}_{A_1}$.

Let $A_2 = A \setminus A_1$. We show that the lifting coefficients of the variables associated with the arcs of A_2 are one. Let $(u,v) \in A_2$ be the first arc in the sequence. Then

$$b_{uv} = p - k - \theta_{(u,v)}(A_1 \cup \{(u,v)\}).$$

Without loss of generality, we may assume that $u = i_{2k+1}$. Note that $v \in V \setminus V_{2k+1}$. First observe that by inequality (3.18) we have $\theta_{(u,v)}(A_1 \cup \{(u,v)\}) \geq p-k-1$. Let $x_2 = x - \sum_{e\in E(V_{2k+1})} \chi_e - \sum_{e\in\delta(u)} x_e\chi_e + \chi_{i_1 i_{2k+2}}$, and $y = \sum_{i\in V_{2k+1}\setminus\{u\}} \gamma_{i0} + \gamma_{uv}$. Then $(x_2, y) \in X_{A_1\cup\{(u,v)\}}$. So $\theta_{(u,v)}(A_1 \cup \{(u,v)\}) = p-k-1$ and thus $b_{uv} = 1$. Therefore, $x(\Delta) + y_{uv} \geq p-k$ is facet defining for $\mathcal{P}_{A_1\cup\{(u,v)\}}$. Let $\overline{A}_2 \subseteq A_2$ be the set of arcs for which the lifting has already been done and $(u,v) \in A_2 \setminus \overline{A}_2$. We assume that $b_a = 1$ for every $a \in \overline{A}_2$ and show that $b_{uv} = 1$. We have

$$b_{uv} = p - k - \theta_{(u,v)}(A_1 \cup \overline{A}_2 \cup \{(u,v)\}).$$

By inequality (3.18), we have $\theta_{(u,v)}(A_1 \cup \overline{A}_2 \cup \{(u,v)\}) \geq p-k-1$. In addition,

$(x_2, y) \in X_{A_1 \cup \overline{A}_2 \cup \{(u,v)\}}$. So $\theta_{(u,v)}(A_1 \cup \overline{A}_2 \cup \{(u,v)\}) = p - k - 1$ and $b_{uv} = 1$. Therefore $x(\Delta) + \sum_{l \in L} \sum_{j \in V \setminus V_l} y_{i_l j} \geq p - k$ is facet defining for $\mathcal{P}_A$. $\square$

3.2.6 Star-path inequalities

Remember the clique constraints (3.10) of the formulation. By generalizing the clique inequalities, another class of valid inequalities is obtained. These inequalities will be referred to as the star-path inequalities. Providing an example of a fractional solution and a star-path inequality that cuts off this solution will be useful.

Let $V = \{0, \ldots, 5\}$. Consider the fractional point $(\overline{x}, \overline{y})$ depicted in Figure 3.3. Again the edges and arcs with value equal to 0 are omitted. The remaining edges and arcs have values 0.5 or 1. The nodes i for which $\overline{y}_{ii} = 1$ are represented by rectangles, those with $\overline{y}_{ii} = 0.5$ are represented by triangles, and finally the nodes with $\overline{y}_{ii} = 0$ are represented by ellipses. The backbone edges with value 1 are represented by bold lines and those with value 0.5 are represented by dashed lines. We use dashed lines with arrows for the assignment arcs.

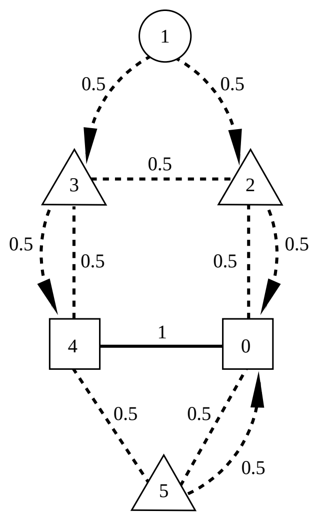

Figure 3.3: The backbone edges and assignment arcs in the fractional solution $(\overline{x}, \overline{y})$.

Consider nodes 1, 2, and 3. Notice that for $(x, y) \in X$, if no concentrators are installed at nodes 2 and 3, i.e., $\sum_{j\in V\setminus\{2\}} y_{2j} = 1$ and $\sum_{j\in V\setminus\{3\}} y_{3j} = 1$, then the edges $\{1,2\}$ and $\{2,3\}$ cannot be used in the backbone network and node 1 cannot be assigned to either of nodes 2 or 3, hence we must have $x_{12}+x_{23}+y_{12}+y_{13} = 0$. If a concentrator is installed at node 3 but not at node 2, i.e., $\sum_{j\in V\setminus\{2\}} y_{2j} = 1$ and $\sum_{j\in V\setminus\{3\}} y_{3j} = 0$, then as the edges $\{1,2\}$ and $\{2,3\}$ cannot be in the backbone and node 1 cannot be assigned to node 2, we have $x_{12}+x_{23}+y_{12} = 0$ and y_{13} can be 0 or 1. If the opposite happens, i.e., $\sum_{j\in V\setminus\{2\}} y_{2j} = 0$ and $\sum_{j\in V\setminus\{3\}} y_{3j} = 1$, then $x_{23}+y_{13} = 0$ and x_{12} and y_{12} can be 0 or 1, but we must have $x_{12}+y_{12} \le 1$. Finally, if concentrators are installed at both nodes 2 and 3, i.e., $\sum_{j\in V\setminus\{2\}} y_{2j} = 0$ and $\sum_{j\in V\setminus\{3\}} y_{3j} = 0$, then $x_{12} + y_{12} + y_{13} \le 1$ and x_{23} can be 0 or 1. So the inequality

$$x_{12} + x_{23} + y_{12} + y_{13} + \sum_{j\in V\setminus\{2\}} y_{2j} + \sum_{j\in V\setminus\{3\}} y_{3j} \le 2 \qquad (3.19)$$

is valid for $\mathcal{P}$.

Now notice that $\overline{x}_{12}+\overline{x}_{23}+\overline{y}_{12}+\overline{y}_{13}+\sum_{j\in V\setminus\{2\}} \overline{y}_{2j}+\sum_{j\in V\setminus\{3\}} \overline{y}_{3j} = 2.5 > 2$. Therefore, adding this inequality to the formulation cuts off the fractional solution $(\overline{x}, \overline{y})$.

We remark here that x_{23}, y_{24}, y_{12}, y_{13}, y_{34} form an odd hole of size 5 in the conflict graph associated with X_S. Thus the odd hole inequality $x_{23} + y_{24} + y_{12} + y_{13} + y_{34} \le 2$ is valid for X and this odd hole inequality is violated by $(\overline{x}, \overline{y})$. Inequality (3.19) can be obtained by lifting this odd hole inequality sequentially with y_{2j} for $j \in V \setminus \{2,4\}$, y_{3j} for $j \in V \setminus \{3,4\}$ and x_{12}.

Now we can provide the general form of these inequalities. Let $m \ge 1$ be an integer and $I_m = \{i_0, \ldots, i_m\}$ be an ordered subset of $V \setminus \{0\}$ consisting of distinct nodes. Let $P_I = \{\{i_l, i_{l+1}\} \in E : i = 0, \ldots, m-1\}$. Note that P_I is a path between i_0 and i_m. Consider the inequality

$$x(P_I) + \sum_{i \in I \setminus \{i_0\}} \sum_{(i,j) \in A} y_{ij} + \sum_{j \in I:(i_0,j) \in A} y_{i_0 j} \leq m \tag{3.20}$$

Clearly, inequality (3.19) is a special case of a more general family of valid inequalities given in inequality (3.20). In the next theorem, we show the validity of inequalities (3.20).

Theorem 3.11 *Inequality (3.20) is valid for $\mathcal{P}$.*

Proof We will prove the validity by induction on $m = |P_{I_m}|$. If $m = 1$, the star-path inequality reduces to

$$x_{i_0 i_1} + \sum_{j \in V \setminus \{i_1\}} y_{i_1 j} + y_{i_0 i_1} \leq 1$$

which is nothing but constraint (3.10) for (i_0, i_1) and hence it is valid for $\mathcal{P}$.

Now assume that the star-path inequalities are valid for $m \leq k$. By the induction hypothesis,

$$x(P_{I_k}) + \sum_{l=1}^{k} \sum_{j \in V \setminus \{i_l\}} y_{i_l j} + \sum_{l=1}^{k} y_{i_0 i_l} \leq k$$

holds for any $(x, y) \in \mathcal{P}$. If $x_{i_k i_{k+1}} + \sum_{j \in V \setminus \{i_{k+1}\}} y_{i_{k+1} j} + y_{i_0 i_{k+1}} \leq 1$, then summing this with the above inequality gives $x(P_{I_{k+1}}) + \sum_{l=1}^{k+1} \sum_{j \in V \setminus \{i_l\}} y_{i_l j} + \sum_{l=1}^{k+1} y_{i_0 i_l} \leq k + 1$.

If $x_{i_k, i_{k+1}} + \sum_{j \in V \setminus \{i_{k+1}\}} y_{i_{k+1} j} + y_{i_0 i_{k+1}} \geq 2$, then, as we know that $x_{i_k, i_{k+1}} + \sum_{j \in V \setminus \{i_{k+1}\}} y_{i_{k+1} j} \leq 1$ and $\sum_{j \in V \setminus \{i_{k+1}\}} y_{i_{k+1} j} + y_{i_0 i_{k+1}} \leq 1$, $x_{i_k, i_{k+1}} = 1$, $\sum_{j \in V \setminus \{i_{k+1}\}} y_{i_{k+1} j} = 0$, and $y_{i_0 i_{k+1}} = 1$. This implies that $x_{i_0 i_1} = 0$ and $\sum_{l=1}^{k+1} y_{i_0 i_l} = 1$. Moreover we have that the inequalities $x_{i_l, i_{l+1}} + \sum_{j \in V \setminus \{i_l\}} y_{i_l j} \leq 1$ are valid for $l = 1, \ldots, k$. Summing up these inequalities together with $x_{i_0 i_1} = 0$, $\sum_{l=1}^{k+1} y_{i_0 i_l} = 1$ and $\sum_{j \in V \setminus \{i_{k+1}\}} y_{i_{k+1} j} = 0$ yields $x(P_{I_{k+1}}) + \sum_{l=1}^{k+1} \sum_{j \in V \setminus \{i_l\}} y_{i_l j} + \sum_{l=1}^{k+1} y_{i_0 i_l} \leq k + 1$. Thus, inequality (3.20) is valid for $\mathcal{P}$. $\square$

Inequalities of type (3.20) will be called *star-path inequalities*. Observe that inequalities (3.10) represent a special case of star-path inequalities. Moreover by Corollary 3.1, the former ones are facet defining for $\mathcal{P}$. Now we also show that the star-path inequalities also define facets for $\mathcal{P}$.

Theorem 3.12 *If $|V \setminus I| \geq 3$, then inequality (3.20) is facet defining for $\mathcal{P}$.*

Proof Let $\mathcal{F} = \{(x,y) \in \mathcal{P} : x(P_I) + \sum_{l=1}^{m} \sum_{j \in V\setminus\{i_l\}} y_{i_l j} + \sum_{l=1}^{m} y_{i_0 i_l} = m\}$. Assume that every solution $(x,y) \in \mathcal{F}$ also satisfies $ax + by = \beta$. For $l = 0,\dots,m$, define $V_{0l} = \{i_0,\dots,i_l\}$, $V_{lm} = \{i_l,\dots,i_m\}$, $x^l = \sum_{e\in E(V\setminus V_{lm})} \chi_e$, $\overline{x}^l = \sum_{e \in E(V\setminus V_{0l})} \chi_e$.

Let $x = \sum_{e\in E} \chi_e$. The solution $(x, 0)$ is in $\mathcal{F}$. Let $e \in E \setminus P_I$. As the solution $(x - \chi_e, 0)$ is also in $\mathcal{F}$, we have $a_e = 0$.

Let $j \in V \setminus V_{0m}$ and $k \in V \setminus \{j\}$. The solution $(x - \sum_{e\in\delta(j)} \chi_e, \gamma_{jk})$ is also in $\mathcal{F}$ and hence $b_{jk} = 0$.

Let $k \in V \setminus \{i_m\}$. As both solutions $(x, 0)$ and $(x^m, \gamma_{i_m k})$ are in $\mathcal{F}$ and $a_e = 0$ for all $e \in E \setminus P_I$, we have $a_{i_{m-1}i_m} = b_{i_m k} = \sigma_m$ for all $k \in V \setminus \{i_m\}$ for some $\sigma_m \in \mathbb{R}$.

Let $l \in \{1,\dots,m-1\}$ and $k \in V \setminus V_{lm}$. As both solutions $(x^{l+1}, \sum_{j=l+1}^{m} \gamma_{i_j 0})$ and $(x^l, \sum_{j=l+1}^{m} \gamma_{i_j 0} + \gamma_{i_l k})$ are in $\mathcal{F}$, we can conclude that $a_{i_{l-1},i_l} = b_{i_l k} = \sigma_l$ for all $k \in V \setminus V_{lm}$ for some $\sigma_m \in R$..

Let $k \in V \setminus V_{0m}$. As both solutions $(x^1, \sum_{j=1}^{m} \gamma_{i_j 0})$ and $(x^0, \sum_{j=1}^{m} \gamma_{i_j 0} + \gamma_{i_0 k})$ are both in $\mathcal{F}$, we can conclude that $b_{i_0 k} = 0$ for all $k \in V \setminus V_{0m}$.

Let $k \in V_{1m}$. Consider the solutions $(x, 0)$ and $(\overline{x}^0, \gamma_{i_0,k})$. As both of these solutions are in $\mathcal{F}$, we have $b_{i_0 k} = a_{i_0,i_1} = \sigma_1$.

Let $l \in \{1,\dots,m-1\}$ and $k \in V_{l+1,m}$. Solutions $(\overline{x}^{l-1}, \sum_{j\in V_{0l-1}} \gamma_{i_j,i_m})$ and $(\overline{x}^l, \sum_{j \in V_{0l-1}} \gamma_{i_j,i_m} + \gamma_{i_l k})$ are both in $\mathcal{F}$, and hence $a_{i_l,i_l+1} = b_{i_l,k} = \sigma_l$ for all $k \in V_{l+1,m}$.

Now as $a_{i_{l-1},i_l} = \sigma_l = a_{i_l,i_l+1}$ for all $l \in \{1,\dots,m-1\}$, we have $\sigma_l = \sigma$ for all $l \in \{1,\dots,m\}$.

This proves that $ax + by = \beta$ is a multiple of $x(P_I) + \sum_{l=1}^{m}\sum_{j\in V\setminus\{i_l\}} y_{i_l j} + \sum_{l=1}^{m} y_{i_0 i_l} = m$. $\square$

3.2.7 A Cut Based Valid Inequality

We end this chapter with a valid inequality based on the triangle and cut inequalities. Consider the triangle inequality $y_{ij} + y_{jk} + y_{ki} \leq 1$ defined by three distinct nodes, $i, j, k \in V \setminus \{0\}$. Let $S \subseteq V \setminus \{0\}$ such that $i, j, k \in S$. Clearly if the left hand side of the triangle inequality is equal to one, this means that there is at least one regenerator in S. So we can impose that at least two edges should be used from $\delta(S)$ and we can construct a cut based valid inequality as follows.

Let $S \subseteq V\setminus\{0\}$ and i, j, k be distinct nodes of S, then the following inequality is valid for $\mathcal{F}$.

$$x(\delta(S)) \geq 2(y_{ij} + y_{jk} + y_{ki}) \qquad S \subseteq V \setminus \{0\},\ i, j, k \in S \tag{3.21}$$

Labbé et al. [30] show that inequality (3.21) is facet defining for the polytope associated with the RSP and we also show that inequality (3.21) defines a facet of $\mathcal{P}$, as well.

Theorem 3.13 *Let $S \subseteq V \setminus \{0\}$ and $i, j, k \in S$ be distinct nodes. If $|S| \geq 4$ and $|V \setminus S| = 1$ or $|V \setminus S| \geq 3$ then inequality (3.21) defines a facet of $\mathcal{P}$.*

Proof Let $\mathcal{F} = \{(x, y) \in \mathcal{P} : x(\delta(S)) = 2(y_{ij} + y_{jk} + y_{ki})\}$. Suppose that every solution (x, y) in $\mathcal{F}$ also satisfies $ax + by = \beta$. We will show that $ax + by = \beta$ is a multiple of $x(\delta(S)) = 2(y_{ij} + y_{jk} + y_{ki})$.

Let $e_1, e_2 \in \delta(S)\setminus\delta(i)$ and $x' = \sum_{e\in(E(S)\cup E(V\setminus S))\setminus\delta(i)} \chi_e$. Consider the solution $(x' + \chi_{e_1} + \chi_{e_2}, \gamma_{ij})$. Clearly this solution is in $\mathcal{F}$. Let $e_3 \in \delta(S) \setminus (\delta(i) \cup \{e_1, e_2\})$.

It can be seen that $(x' + \chi_{e_1} + \chi_{e_3}, \gamma_{ij})$ and $(x' + \chi_{e_2} + \chi_{e_3}, \gamma_{ij})$ are also solutions in $\mathcal{F}$. This shows that $a_e = \sigma$ for some $\sigma \in \mathbb{R}$ for all $e \in \delta(S) \setminus \delta(i)$. Now let $e_1, e_2, e_3 \in \delta(S) \setminus \delta(j)$ and $x^{''} = \sum_{e \in (E(S) \cup E(V \setminus S)) \setminus \delta(j)} \chi_e$. Consider the similar solutions $(x^{''} + \chi_{e_1} + \chi_{e_2}, \gamma_{jk})$, $(x^{''} + \chi_{e_1} + \chi_{e_3}, \gamma_{jk})$, and $(x^{''} + \chi_{e_2} + \chi_{e_3}, \gamma_{jk})$. Since all three solutions are in $\mathcal{F}$, we can conclude that $a_e = \sigma$ for all $e \in \delta(S)$.

To find the coefficients of the edges in $E(S)$, let $\{u, v\} \in E(S) \setminus \delta(i)$ and $e_1, e_2 \in \delta(S) \setminus \delta(i)$ such that $u \in e_1$ and $v \in e_2$. We can see that $(x' + \chi_{e_1} + \chi_{e_2}, \gamma_{ij})$ is a solution in $\mathcal{F}$. As $(x' + \chi_{e_1} + \chi_{e_2} - \chi_{uv}, \gamma_{ij})$ is also in $\mathcal{F}$, $a_e = 0$ for all $e \in E(S) \setminus \delta(i)$. In addition, we can easily extend this result. Let $\{i, u\} \in E(S) \setminus \delta(j)$, $e_1, e_2 \in \delta(S) \setminus \delta(j)$ such that $i \in e_1$ and $u \in e_2$. Since $(x^{''} + \chi_{e_1} + \chi_{e_2}, \gamma_{jk})$ and $(x^{''} + \chi_{e_1} + \chi_{e_2} - \chi_{iu}, \gamma_{jk})$ are both in $\mathcal{F}$, we can say that $a_e = 0$ for all $e \in E(S) \setminus \{\{i, j\}\}$. For the remaining edge $\{i, j\}$, let $e_1, e_2 \in \delta(S) \setminus \delta(k)$ such that $i \in e_1$ and $j \in e_2$. We also define $x^{'''} = \sum_{e \in (E(S) \cup E(V \setminus S)) \setminus \delta(k)} \chi_e$. It can be easily seen that $(x^{'''} + \chi_{e_1} + \chi_{e_2}, \gamma_{ki})$ and $x^{'''} + \chi_{e_1} + \chi_{e_2} - \chi_{ij}, \gamma_{ki})$ are both in $\mathcal{F}$. Therefore, $a_e = 0$ for all $e \in E(S)$.

The case for the edges in $V \setminus S$ is simpler. If $|V \setminus S| = 1$ there is no edge in $V \setminus S$, so we assume $|V \setminus S| \geq 3$. Let $\{u, v\} \in E(V \setminus S)$, $e_1, e_2 \in \delta(S)$ such that $u \in e_1$ and $v \in e_2$. Clearly, $(x' + \chi_{e_1} + \chi_{e_2}, \gamma_{ij})$ and $(x' + \chi_{e_1} + \chi_{e_2} - \chi_{uv}, \gamma_{ij})$ are both solutions in $\mathcal{F}$. So we conclude that, $a_e = 0$ for all $e \in E(V \setminus S)$.

Let $l \in V \setminus \{0, i, j, k\}$ and $u, v, w \in V \setminus \{l, i\}$. Let e_1, e_2 be two edges in $\delta(S) \setminus \delta(i)$. Defining $\overline{x} = \sum_{e \in (E(S) \cup E(V \setminus S)) \setminus (\delta(i) \cup \delta(l))} \chi_e$ we can find the solution $(\overline{x} + \chi_{e_1} + \chi_{e_2}, \gamma_{ij} + \gamma_{lu})$ in $\mathcal{F}$. Observe that the solution $(\overline{x} + \chi_{e_1} + \chi_{e_2}, \gamma_{ij} + \gamma_{lv})$ is also in $\mathcal{F}$. Moreover, $(x'' + \chi_{e_1} + \chi_{e_2}, \gamma_{jk})$ is in $\mathcal{F}$. So we have $b_{uv} = 0$ for all $(u, v) \in A$ such that $u \in V \setminus \{0, i, j, k\}$.

Similarly, let $u, v, w \in V \setminus \{i, j\}$ and define $\hat{x} = \sum_{e \in (E(S) \cup E(V \setminus S)) \setminus (\delta(i) \cup \delta(j))} \chi_e$. Clearly, $(\hat{x} + \chi_{e_1} + \chi_{e_2}, \gamma_{iu} + \gamma_{jk})$ is a solution in $\mathcal{F}$ where $e_1, e_2 \in \delta(S) \setminus (\delta(i) \cup \delta(j))$. Observe that the solution $(\hat{x} + \chi_{e_1} + \chi_{e_2}, \gamma_{iv} + \gamma_{jk})$ is also in $\mathcal{F}$. Moreover, $(x'' + \chi_{e_1} + \chi_{e_2}, \gamma_{jk})$ is also a solution in $\mathcal{F}$. These solutions can be constructed for nodes j and k in a similar way. So we have $b_{uv} = 0$ for all $(u, v) \in A \setminus \{(i, j), (j, k), (k, i)\}$.

Finally, let $e_1, e_2 \in \delta(S) \setminus (\delta(i) \cup \delta(j) \cup \delta(k))$. Clearly, the solutions $(x' + \chi_{e_1} +$

$\chi_{e_2}, \gamma_{ij})$, $(x'' + \chi_{e_1} + \chi_{e_2}, \gamma_{jk})$, and $(x''' + \chi_{e_1} + \chi_{e_2}, \gamma_{ki})$ are all in $\mathcal{F}$. Considering these solutions together with the solution $(\sum_{e \in E(V \setminus S)} \chi_e, \sum_{u \in S} \gamma_{u0})$, we can see that $b_a = -2\sigma$ for $a \in \{(i,j), (j,k), (k,i)\}$.

Therefore, $ax + by = \beta$ is a multiple of $y_{ij} + y_{jk} + y_{ki} = 1$ and $\mathcal{F}$ is a facet of $\mathcal{P}$. $\square$

3.3 Conclusion

The 2ECSSP is studied in this chapter and a mathematical formulation is proposed for the problem. Some valid inequalities are described to improve the mathematical formulation. The polytope associated with the formulation is analyzed and the conditions for the inequalities under which they are facet defining are discussed. The aim of this chapter is to identify the polyhedral structure of the problem and we focus on the solution technique for the problem in the next chapter.

Although the results of this chapter are developed for the 2ECSSP in which the backbone network is 2-edge connected, it can be shown that the results can be extended to the kECSSP which has a k-edge connected backbone network. The mathematical formulation for the kECSSP will be the same except the coefficients of the y variables in the cut inequalities will be k instead of 2. The extended F-partition and star-path inequalities will be still valid. However, some coefficients need to be modified in the extended F-partition inequalities. Note that $|F|$ must be odd since otherwise the extended F-partition inequalities are implied by the cut and trivial inequalities. This condition is sufficient if k is even, however, when k is odd then we need to impose additional conditions to ensure that extended F-partition inequalities are not implied by other constraints. However, the conditions for the valid inequalities to be facet defining may change so the polyhedral analysis should be extended to see if the valid inequalities are still facet defining.

In addition, these are not the only valid inequalities for the kECSSP with

$k \geq 3$. There may be other valid inequalities which could be very effective in the solution of the kECSSP. So we can say that the polyhedral analysis of the 2ECSSP provides only a basis for the analysis of the kECSSP.

Chapter 4

A Branch and Cut Algorithm for the Hierarchical Survivable Network Design Problem with Single Homing

In the previous chapter, we discussed the 2ECCSP, proposed an integer linear formulation and some valid inequalities. A polyhedral analysis was performed and the conditions under which the inequalities define facets were described. As the number of constraints of the proposed mathematical formulation is exponential it is not possible to solve the model directly. For this reason, a branch-and-cut algorithm, in which the constraints are added to the model as they are needed, is required. The development of such an algorithm includes defining the separation problems that will be used to identify violated inequalities. In this chapter, we will describe the separation problems, and some operations called *reduction operations* which are proposed to reduce the dimensions of the separation problems and hence making them easier in practice. We are also going to provide the implementation details of the branch-and-cut algorithm and discuss the computational results obtained by using the proposed algorithm. Since the reduction operations are used in the separation problems we start with the description of the reduction operations.

4.1 Reduction Operations

The term reduction is used because applying these operations reduces the size of the problem that must be solved. The reduction operations use ideas developed by Fonlupt and Mahjoub [12] for the 2-edge connected subgraph polytope.

Remember we have an undirected graph $G = (V, E)$ and a directed graph $D = (V, A)$ which are both associated with the 2ECSSP. The reduction operations may affect both. We shall first introduce some notation and concepts, before describing these operations.

Given $e = uv \in E$, contracting e means deleting e from E and arcs $(u, v), (v, u)$ from A, identifying u and v, deleting the resulting loops, and keeping the new parallel edges and arcs. Similarly contracting a set of nodes $W \subset V$ means deleting set of edges $E(W)$ and set of arcs $A(W)$, identifying W as a single node, deleting the resulting loops and keeping the new parallel edges and arcs.

We will denote by $Q(G)$ the polytope given by inequalities (3.10) - (3.14). That is to say, $Q(G)$ is the linear relaxation of 2ECSSP(G). Clearly, $Q(G)$ is defined in terms of both graphs G and D. However, as G and D are closely related, we will only write $Q(G)$ for $Q(G, D)$. If $(\overline{x}, \overline{y})$ is a solution of $Q(G)$ we will denote by $E_0(\overline{x}), E_1(\overline{x})$, and $E_f(\overline{x})$, the set of edges $e \in E$ such that $\overline{x}(e) = 0, \overline{x}(e) = 1$, and $0 < \overline{x}(e) < 1$, respectively. Similarly, we will denote by $A_0(\overline{y}), A_1(\overline{y})$, and $A_f(\overline{y})$ the set of arcs $a \in A$ with $\overline{y}(a) = 0, \overline{y}(a) = 1$, and $0 < \overline{y}(a) < 1$, respectively. We also use $\Gamma(\overline{x}, \overline{y}), T(\overline{x}, \overline{y}), \xi(\overline{x}, \overline{y})$ to denote the set of arcs of A, nodes of $V \setminus \{0\}$, and pairs (S, i) for all $S \subseteq V \setminus \{0\}, i \in S$, respectively, for which the corresponding inequalities (3.10),(3.11), and (3.12) are tight for $(\overline{x}, \overline{y})$.

Let $(\overline{x}, \overline{y})$ be an extreme point of $Q(G)$. Thus there is a set of arcs $\Gamma^*(\overline{x}, \overline{y}) \subseteq \Gamma(\overline{x}, \overline{y})$, a set of nodes $T^*(\overline{x}, \overline{y}) \subseteq T(\overline{x}, \overline{y})$ and a set $\xi^*(\overline{x}, \overline{y}) \subseteq \xi(\overline{x}, \overline{y})$ such that

$(\overline{x}, \overline{y})$ is the unique solution of the system

$$R(\overline{x},\overline{y}) = \begin{cases} x_{ij} + y_{ij} + \sum_{k\in V\setminus\{j\}} y_{jk} = 1 & (i,j) \in \Gamma^*(\overline{x},\overline{y}), \\ x_{0i} + \sum_{k\in V\setminus\{i\}} y_{ik} = 1 & i \in T^*(\overline{x},\overline{y}), \\ x(\delta(S)) + 2\sum_{j\in V\setminus S} y_{ij} = 2 & (S,i) \in \xi^*(\overline{x},\overline{y}), \\ x_{ij} = 0 & ij \in E_0(\overline{x}), \\ x_{ij} = 1 & ij \in E_1(\overline{x}), \\ y_{ij} = 0 & (i,j) \in A_0(\overline{y}), \\ y_{ij} = 1 & (i,j) \in A_1(\overline{y}). \end{cases} \tag{4.1}$$

Note that the nontrivial equations of $R(\overline{x}, \overline{y})$ must have at least two variables with fractional values (note that the right hand side of each of these equations is integer). If all the variables of one of these equations have value 0 or 1, then that inequality would be redundant with respect to $x_{ij} = 0$, $ij \in E_0(\overline{x})$, $x_{ij} = 1$, $ij \in E_1(\overline{x})$, $y_{ij} = 0$, $(i,j) \in A_0(\overline{y})$ and $y_{ij} = 1$, $(i,j) \in A_1(\overline{y})$.

Let $(\overline{x}, \overline{y})$ be a solution of $Q(G)$. Consider the following operations with respect to $(\overline{x}, \overline{y})$:

- θ_1: Delete an edge e with $\overline{x}_e = 0$.
- θ_2: Delete an arc (i,j) with $\overline{y}_{ij} = 0$.
- θ_3: Delete a node i as well as all the edges and arcs incident to it, if there is some j such that $\overline{y}_{ij} = 1$.
- θ_4: Contract a node set W such that $G(W)$ is 2-edge connected and $\overline{x}_e = 1$ for every $e \in E(W)$.
- θ_5: Contract an edge e if at least one of the endpoints of e is incident to exactly two edges, and these two edges have value 1 with respect to $\overline{x}$.

Note that the edges and the arcs with fractional values are preserved by all the reduction operations. Note also that θ_1 and θ_2 modify only G while the remaining ones affect both G and D. Starting from $G = (V, E)$ and $D = (V, A)$ and applying repeatedly $\theta_1, \ldots, \theta_5$, we obtain reduced graphs $G' = (V', E')$, $D' = (V', A')$ and a solution $(\overline{x}', \overline{y}') \in Q(G')$. We remark that $(\overline{x}', \overline{y}')$ is nothing but the restriction of $(\overline{x}, \overline{y})$ in G' and D'. We claim that necessary information is preserved through the reduction operations. The following theorem establishes the relation between the two polytopes.

Theorem 4.1 $(\overline{x}, \overline{y})$ *is an extreme point of* $\mathcal{Q}(G)$ *if and only if* $(\overline{x}', \overline{y}')$ *is an extreme point of* $\mathcal{Q}(G')$*.*

Proof Suppose $(\overline{x}, \overline{y})$ is an extreme point of $Q(G)$. Without loss of generality, we may suppose that $(\overline{x}', \overline{y}')$ is obtained by the application of $\theta_1, \ldots, \theta_5$ exactly once. It is clear that if $(\overline{x}', \overline{y}')$ is obtained by either operation θ_1 or θ_2, then $(\overline{x}', \overline{y}')$ is an extreme point of $Q(G')$. Now suppose that $\overline{y}_{ij} = 1$ and that $(\overline{x}', \overline{y}')$ is obtained by the application of θ_3 with respect to node i. First observe that, by inequalities (3.10), we have $\overline{x}_{il} = 0$ for all $\{i, l\} \in E$, $\overline{y}_{il} = 0$ for all $(i, l) \in A$ with $l \neq i$, and $\overline{y}_{li} = 0$ for all $(l, i) \in A$ with $l \neq i$. Moreover, it is clear that inequalities (3.10) and (3.11) with respect to G' are satisfied by $(\overline{x}', \overline{y}')$. Now consider a cut $\delta(S')$ of G' and a node $k \in S'$. Note that $k \neq i$. As $(\overline{x}, \overline{y}) \in Q(G)$ we have $2 \leq \overline{x}(\delta_G(S')) + 2\sum_{l \in V \setminus S'} \overline{y}_{kl} = \overline{x}(\delta_{G'}(S')) + \sum_{l \in S'} \overline{x}_{il} + 2\sum_{l \in V' \setminus S'} \overline{y}_{kl} + 2\overline{y}_{ki} = \overline{x}'(\delta_{G'}(S')) + 2\sum_{l \in V' \setminus S'} \overline{y}'_{kl}$, and hence the cut inequality (3.12) induced by (S', k) in G' is satisfied by $(\overline{x}', \overline{y}')$. Thus $(\overline{x}', \overline{y}')$ is a solution of $Q(G')$. Moreover, all the edges and arcs removed from the graph have integer values. So, they appear as trivial equations in system $R(\overline{x}, \overline{y})$. Consequently, $(\overline{x}', \overline{y}')$ is the unique solution of a subsystem of $R(\overline{x}, \overline{y})$, and therefore it is an extreme point of $Q(G')$.

Now suppose $(\overline{x}', \overline{y}')$ comes from the application of θ_4 with respect to a node set W. First note that all the arcs with both endnodes in W have value zero with respect to $\overline{y}$. It is easy to see that inequalities (3.10) and (3.11) remain satisfied by $(\overline{x}', \overline{y}')$ in G'. Let $U' \subseteq V'$ and $k \in U'$. Let w be the node of V' which arises from the contraction of W and, without loss of generality, suppose that $w \in U'$. Let $U = (U' \setminus \{w\}) \cup W$. As $k \in U$ and $(\overline{x}, \overline{y})$ is a solution of $Q(G)$, we have $2 \leq \overline{x}(\delta_G(U)) + 2\sum_{l \in V \setminus U} \overline{y}_{kl} = \overline{x}(\delta_{G'}(U')) + 2\sum_{l \in V' \setminus U'} \overline{y}_{kl}$, and hence the cut inequality (3.12) induced by (U', k) in G' is satisfied by $(\overline{x}', \overline{y}')$. Therefore $(\overline{x}', \overline{y}')$ is a solution of $Q(G')$.

Now suppose, on the contrary, that $(\overline{x}', \overline{y}')$ is not an extreme point of $Q(G')$. Thus there exist two solutions $(\overline{x}^{1'}, \overline{y}^{1'})$ and $(\overline{x}^{2'}, \overline{y}^{2'})$ of $Q(G')$ such that $(\overline{x}', \overline{y}') = \frac{1}{2}((\overline{x}^{1'}, \overline{y}^{1'}) + (\overline{x}^{2'}, \overline{y}^{2'}))$. Consider the solution given by

$$\overline{x}^i_e = \begin{cases} \overline{x}^{i'}_e, & \text{for all } e \in E \setminus E(W) \\ 1, & \text{for all } e \in E(W) \end{cases}$$

and

$$\overline{y}^i_a = \begin{cases} \overline{y}^{i'}_a, & \text{for all } a \in A' \\ 0, & \text{otherwise} \end{cases}$$

for $i = 1, 2$. Clearly, $(\overline{x}^i, \overline{y}^i) \in Q(G)$ for $i = 1, 2$. Moreover, $(\overline{x}, \overline{y}) = \frac{1}{2}((\overline{x}^1, \overline{y}^1) + (\overline{x}^2, \overline{y}^2))$. This contradicts the extremality of $(\overline{x}, \overline{y})$. The proof is similar for θ_5. Repeating a similar line of arguments, one can easily show the converse. □

Theorem 4.1 is important from an algorithmic point of view. It shows the correspondence between the extreme points of $Q(G)$ and those of $Q(G')$. This shows that any algorithm used for separating fractional extreme points of $Q(G')$ may also be used for separating the corresponding fractional extreme points of $Q(G)$.

Besides, it will be also important if we can show that if there is a violated inequality of a particular class, then it can be identified in the reduced space. In the following theorems, we show that there is a strong relation between the violated inequalities in the original and reduced spaces. This will compose the algorithmic consequences of the reduction operations.

Theorem 4.2 *There is a cut inequality (3.12) violated by $(\overline{x}, \overline{y})$ in G if and only if there is a cut inequality violated by $(\overline{x}', \overline{y}')$ in G'.*

Proof Let $S \subseteq V$, $i \in S$ and suppose that the cut inequality induced by (S, i) is violated by $(\overline{x}, \overline{y})$, that is to say $\overline{x}(\delta(S)) + 2\sum_{j \in V \setminus S} \overline{y}_{ij} < 2$. We will show that there is a node set $S' \subseteq V'$ and a node $i' \in S'$ whose corresponding cut inequality in G' is violated by $(\overline{x}', \overline{y}')$. First, it is clear that if $(\overline{x}', \overline{y}')$ is obtained by operation θ_1 (resp. θ_2) with respect to an edge ij (resp. arc (i, j)) such that $\overline{x}_{ij} = 0$ (resp. $\overline{y}_{ij} = 0$), then the same inequality is violated by $(\overline{x}', \overline{y}')$ in G'. Suppose $(\overline{x}', \overline{y}')$ is obtained by θ_3 with respect to an arc (u, w) with $\overline{y}_{uw} = 1$. By inequality (3.10), we have $\overline{x}_{uv} = 0$ for all $v \in V \setminus \{u\}$, $\overline{y}_{uv} = 0$ for all $v \in V \setminus \{u, w\}$ and $\overline{y}_{wv} = 0$ for all $v \in V \setminus \{w\}$. If $u \neq i$, then $(S \setminus \{u\}, i)$ (resp. (S, i)) induces a violated cut inequality with respect to $(\overline{x}, \overline{y})$, if $u \in S$ (resp $u \notin S$). That is to say we can take $S' = S$ and $i' = i$ if $u \notin S$, and $S' = S \setminus \{u\}$ and $i' = i$ if $u \in S$. If $u = i$, as $\overline{y}_{uw} = 1$ and the cut inequality induced by (S, u) is violated, it follows that $w \in S$. By considering $S' = S \setminus \{u\}$ and $i' = w$, we have that the cut inequality in G' induced by (S', i') is violated.

Suppose $(\overline{x}', \overline{y}')$ is obtained by θ_4 with respect to a node set $W \subset V$. Let w be the node that arises from the contraction of W. Since $(W, E(W))$ is 2-edge connected and $\overline{x}_e = 1$ for all $e \in E(W)$, we should have either $W \subseteq S$ or $W \subseteq V \setminus S$, for otherwise, the cut inequality induced by (S, i) would not be violated.

If $W \subseteq S$, then set $S' = (S \setminus W) \cup \{w\}$ and $i' = w$ (resp. $i' = i$), if $i \in W$ (resp. $i \in S \setminus W$).

If $W \subseteq V \setminus S$, then set $S' = S$ and $i' = i$.

In both cases, (S', i') induces a cut inequality in G' which is violated by $(\overline{x}', \overline{y}')$.

Finally, suppose $(\overline{x}', \overline{y}')$ is obtained by θ_5 with respect to two edges uv and vw with $\overline{x}_{uv} = \overline{x}_{vw} = 1$ and v with degree two. As the cut induced by (S, i) is violated by $(\overline{x}, \overline{y})$, at most one of the edges uv and vw can be in $\delta(S)$. Suppose, without loss of generality, that $uv \in \delta(S)$, $u \in S, v \in V \setminus S$ and $i \neq u$. Set $S' = (S \setminus \{u\}) \cup \{\overline{v}\}$

and $i' = i$ where $\overline{v}$ is the node that arises from the contraction of uv. We have $\overline{x}(\delta(S)) + 2\sum_{j\in V\setminus S}\overline{y}_{ij} = \overline{x}'(\delta(S')) + 2\sum_{j\in V\setminus S'}\overline{y}'_{ij} < 2$. Therefore, (S', i') induces a violated cut inequality.

Conversely, let $S' \subseteq V'$ and $i' \in S'$ be such that the cut inequality induced by (S', i') is violated by $(\overline{x}', \overline{y}')$. If $(\overline{x}', \overline{y}')$ is obtained by either θ_1, θ_2 or θ_3 then by setting $S = S'$ and $i = i'$ we have that the cut induced by (S, i) is violated by $(\overline{x}, \overline{y})$.

Suppose $(\overline{x}', \overline{y}')$ is obtained by θ_4 with respect to a node set $W \subseteq V \setminus \{0\}$. Let w be the node arising from the contraction of W. If $w \in S'$ and $i' \neq w$ (resp. $i' = w$), then let $S = (S' \setminus \{w\}) \cup W$ and $i = i'$ (resp. $i = v$ for some $v \in W$).

If $w \notin S'$, let $S = S'$ and $i = i'$.

In both cases, the cut induced by (S, i) in G is violated by $(\overline{x}, \overline{y})$.

Finally, suppose $(\overline{x}', \overline{y}')$ is obtained by θ_5 with respect to two edges uv and vw such that $\overline{x}_{uv} = \overline{x}_{vw} = 1$. Let $\overline{v}$ be the node arising from the contraction of uv. If $\overline{v} \notin S'$, we can set $S = S'$ and $i = i'$. If $\overline{v} \in S'$ and $i' = \overline{v}$ (resp. $i \neq \overline{v}$) one can set $S = (S' \setminus \{\overline{v}\}) \cup \{u\}$ and $i = u$ (resp. $i = i'$). In both cases, the cut induced by (S, i) is violated by $(\overline{x}, \overline{y})$.□

The same relation can be shown for the extended F-partition (3.18) inequality. But we give two lemmas that will make the proof easier.

Lemma 4.1 *Let $(V_0, \ldots, V_p)$ be a partition and $e \in \delta(V_0)$ with $\overline{x}_e = 1$. If there is a violated extended F-partition inequality (3.18) for this partition, then there is a violated extended F-partition inequality such that $e \in F$.*

Proof Suppose $(V_0, \ldots, V_p)$ induces a violated extended F-partition inequality, for some $F \subseteq \delta(V_0)$, that is, $\overline{x}(\delta(V_0, \ldots, V_p) \setminus F) + \sum_{l=1}^{p}\sum_{j\in V\setminus V_l}\overline{y}_{i_l j} < p - k$. If $e \in F$, then the lemma holds. So assume that $e \notin F$. Let f be an edge of F. Exchanging f by e in F yields an extended F-partition inequality with a violation not less than the initial one. □

Lemma 4.2 *Let $(V_0, \dots, V_p)$ be a partition and $(u, w) \in A$ with $\overline{y}_{uw} = 1$. If there is a violated extended F-partition inequality (3.18) for this partition, then there is a violated extended F-partition inequality such that $u \notin I = \{i_1, \dots, i_p\}$, i.e., u is not a node fixed in a subset of the partition.*

Proof Suppose $(V_0, \dots, V_p)$ induces a violated extended F-partition inequality, for some $F \subseteq \delta(V_0)$, that is, $\overline{x}(\delta(V_0, \dots, V_p) \setminus F) + \sum_{l=1}^{p} \sum_{j \in V \setminus V_l} \overline{y}_{i_l j} < p - k$. If $u \notin I$, the lemma holds. So assume that $u \in I$. Without loss of generality, assume that $u = i_1$. If $w \in V_1$, then choosing w as the fixed node of V_1, we obtain another violated extended F-partition inequality. So suppose $w \notin V_1$. If $V_1 \setminus \{u\} \neq \emptyset$, then one can choose another node $k \in V_1 \setminus \{u\}$ to be a fixed node in V_1. Note that the left hand side of the extended F-partition inequality will not increase as $\sum_{j \in V \setminus V_1} \overline{y}_{kj} \leq \sum_{j \in V \setminus V_1} \overline{y}_{uj}$. Thus $(V_0, \dots, V_p)$, F, and $(I \setminus \{u\}) \cup \{k\}$ yield another violated extended F-partition inequality. If $V_1 \setminus \{u\} = \emptyset$, then we can put V_1 into V_0. As we know that $\overline{x}(\delta(V_1)) = 0$ due to constraints (3.10), we have $\overline{x}(\delta(V_0, V_2, \dots, V_p) \setminus F) + \sum_{l=2}^{p} \sum_{j \in V \setminus V_l} \overline{y}_{i_l j} < p - k - 1$. So this operation results in a violated extended F-partition inequality. So the lemma holds. $\square$

Theorem 4.3 *There is an extended F-partition inequality (3.18) violated by $(\overline{x}, \overline{y})$ in G if and only if there is an extended F-partition inequality violated by $(\overline{x}', \overline{y}')$ in G'.*

Proof Let $(V_0, \dots, V_p)$ be a partition denoted by P_1, $F \subset \delta(V_0)$ and $I = \{i_1, \dots, i_p\}$ with $i_l \in V_l$ for $l = 1, \dots, p$. Suppose that the extended F-partition inequality induced by P_1, F and I is violated by $(\overline{x}, \overline{y})$, that is to say $\overline{x}(\delta(V_0, \dots, V_p) \setminus F) + \sum_{l=1}^{p} \sum_{j \in V \setminus V_l} \overline{y}_{i_l j} < p - k$. We will show that there is a partition of V', an edge set F' and a node set I' whose corresponding extended F-partition inequality in G' is violated by $(\overline{x}', \overline{y}')$. First observe that if $\overline{x}_e = 0$ for some edge $e \in F$, and the cut inequalities are satisfied by $\overline{x}$, then the extended F-partition inequality cannot be violated by $\overline{x}$. Thus we will suppose, without loss of generality, that $\overline{x}_e > 0$ for all $e \in F$. If $(\overline{x}', \overline{y}')$ is obtained by operation θ_1 with respect to an edge e with $\overline{x}_e = 0$ and $e \notin F$, then the same inequality is violated by $(\overline{x}', \overline{y}')$ in G'. It is clear that if $(\overline{x}', \overline{y}')$ is obtained by operation θ_2

with respect to an arc a such that $\overline{y}_a = 0$, then the same inequality is violated by $(\overline{x}', \overline{y}')$ in G'. Suppose $(\overline{x}', \overline{y}')$ is obtained by θ_3 with respect to an arc (u, w) with $\overline{y}_{uw} = 1$. By Lemma 4.2, we can assume that $u \notin I$. Let $u \in V_k$ for some $k \in \{0, \ldots, p\}$. For $l = 0, \ldots p$, let $V'_l = V_l$ if $k \neq l$ and $V'_l = V_l \setminus \{u\}$ otherwise. It can be seen that $(V'_0, \ldots, V'_p)$, F and I induce an extended F-partition inequality violated by $(\overline{x}', \overline{y}')$ in G'.

Suppose $(\overline{x}', \overline{y}')$ is obtained by θ_4 with respect to a node set $W \subset V$. Let w be the node that arises from the contraction of W. Suppose first $W \subseteq V_k$ for some $k \in \{0, \ldots, p\}$. For $l = 0, \ldots, p$, let $V'_l = V_l$ if $k \neq l$ and $V'_l = (V_l \setminus W) \cup \{w\}$ otherwise, and let $I' = (I \setminus \{i_k\}) \cup \{w\}$ if $k > 0$ and $I' = I$ otherwise. We can see that $(V'_0, \ldots, V'_p)$, F and I' induce an extended F-partition inequality violated by $(\overline{x}', \overline{y}')$ in G'. Suppose now that W is not a subset of a subset of the partition. Without loss of generality, assume that $W \cap V_i \neq \emptyset$ for $i = 1, \ldots, k$. Since $G(W)$ is 2-edge connected and $\overline{x}_e = 1$ for all $e \in E(W)$, $\overline{x}(\delta(V_1, \ldots, V_k)) \geq k$. So we can contract $V_1, \ldots, V_k$, and choose a new fixed node for the resulting set. In this case, the right hand side of the inequality reduces by $k - 1$ while the left hand side decreases by at least k, which yields an extended F-partition inequality violated by $(\overline{x}', \overline{y}')$ in G'. If $W \cap V_0 \neq \emptyset$, then by Lemma 4.1, we can assume that every $e \in \delta(W) \cap \delta(V_0)$ is also in F, implying that $|F \cap E(W)| \geq 2$. Note that $\overline{x}(\delta(V_1, \ldots, V_k)) \geq k-1$. Let $V'_0 = \cup_{i=0}^{k} V_i$, $F' = F \cap \delta(V'_0)$, and $I' = I \setminus \{i_1, \ldots, i_k\}$. Note that $|F'| \leq |F| - 2$. It is not hard to see that the extended F-partition inequality induced by $(V'_0, V_{k+1}, \ldots, V_p)$, F', and I' is violated by $(\overline{x}', \overline{y}')$ in G'.

Finally, suppose $(\overline{x}', \overline{y}')$ is obtained by θ_5 with respect to two edges uv and vw with $\overline{x}_{uv} = \overline{x}_{vw} = 1$ and v with degree two. Let $\overline{v}$ be the node that arises from the contraction of uv. Suppose $uv \in E(V_k)$ for some $k \in \{0, \ldots, p\}$ (the case $vw \in E(V_k)$ is similar). For $l = 0, \ldots, p$, let $V'_l = V_l$ if $k \neq l$ and $V'_l = (V_l \setminus \{u, v\}) \cup \{\overline{v}\}$ otherwise and let $I' = (I \setminus \{i_k\}) \cup \{\overline{v}\}$ if $k > 0$ and $I' = I$ otherwise. It is easily seen that $(V'_0, \ldots, V'_p)$, F and I' induce an extended F-partition inequality violated by $(\overline{x}', \overline{y}')$ in G'. If u, v, w are in different subsets or if $u, w \in V_j$ and $v \in V_k$ with $j \neq k$, then by contracting the subsets intersecting $\{u, v, k\}$ and choosing a new fixed node for the new set we obtain a violated extended F-partition inequality. Therefore, we do not need to consider such cases.

Conversely, let $(V'_0, \ldots, V'_p)$ be a partition of V' denoted by P'_1, $F' \subset \delta(V'_0)$ and $I' = \{i'_1, \ldots, i'_p\}$ with $i'_l \in V'_l$ for $l = 1, \ldots, p$. Suppose that the extended F-partition inequality induced by P'_1, F' and I' is violated by $(\overline{x}', \overline{y}')$, that is to say $\overline{x}'(\delta(V'_0, \ldots, V'_p) \setminus F') + \sum_{l=1}^{p} \sum_{j \in V' \setminus V'_l} \overline{y}'_{i'_l j} < p - k$. If $(\overline{x}', \overline{y}')$ is obtained by either θ_1 or θ_2 then P'_1, F', and I' also induce an extended F-partition inequality violated by $\overline{x}, \overline{y}$ in G. If $(\overline{x}', \overline{y}')$ is obtained by θ_3 with respect to an arc (i, j) then $(V'_0 \cup \{i\}, V'_1, \ldots, V'_p)$, F' and I' also induce an extended F-partition inequality violated by $(\overline{x}, \overline{y})$ in G.

Suppose $(\overline{x}', \overline{y}')$ is obtained by θ_4 with respect to a node set W. Let w be the node arising from the contraction of W and assume that $w \in V'_k$ for some $k \in \{0, \ldots, p\}$. Then $(V_0, \ldots, V'_p)$, F' and I' induce an extended F-partition inequality violated by $(\overline{x}, \overline{y})$ in G, where $V_i = V'_i$ if $i \neq k$, and $V_i = (V'_i \setminus \{w\}) \cup W$ otherwise.

Finally, suppose $(\overline{x}', \overline{y}')$ is obtained by θ_5 with respect to two edges uv and vw such that $\overline{x}_{uv} = \overline{x}_{vw} = 1$. Let $\overline{v}$ be the node arising from the contraction of uv and assume that $\overline{v} \in V'_k$ for some $k \in \{0, \ldots, p\}$. Then $(V_0, \ldots, V_p)$, F' and I' induce an extended F-partition inequality violated by $\overline{x}, \overline{y}$ in G, where $V_i = V'_i$ if $i \neq k$, and $V_i = (V'_i \setminus \{\overline{v}\}) \cup \{u, v\}$ otherwise. $\square$

Finally, we analyze the star-path inequalities (3.20) and start with a lemma.

Lemma 4.3 *An ordered set I cannot yield a violated star-path (3.20) inequality if there is an edge $e \in P_I$ with $\overline{x}_e = 1$.*

Proof Let $I = \{i_0, \ldots, i_m\}$ and $e = \{i_l, i_l + 1\}$. Assume that $e \in P_I$ and $\overline{x}_e = 1$. This implies that $y_{i_l j} = 0$ for all $j \in V \setminus \{i_l\}$ and $y_{i_{l+1}, j} = 0$ for all $j \in V \setminus \{i_{l+1}\}$. Therefore, $\sum_{j \in V \setminus \{i_l\}} y_{i_l j} + \sum_{j \in V \setminus \{i_{l+1}\}} y_{i_{l+1} j} = 0$. As $(\overline{x}, \overline{y})$ satisfies inequalities (3.10) the following hold:

$$\begin{aligned}
&x_{i_0i_1} + \sum_{j=1}^{m} y_{i_0i_j} \leq \; x_{i_0i_1} + y_{i_1i_0} + \sum_{k\in V\setminus\{i_0\}} y_{i_0k} \; \leq 1\\
&\vdots\\
&x_{i_{l-1}i_l} + \sum_{j\in V\setminus\{i_{l-1}\}} y_{i_{l-1}j} \leq \; x_{i_{l-1}i_l} + y_{i_li_{l-1}} + \sum_{j\in V\setminus\{i_{l-1}\}} y_{i_{l-1}j} \leq 1\\
&x_{i_li_{l+1}} + \sum_{j\in V\setminus\{i_l\}} y_{i_lj} + \sum_{j\in V\setminus\{i_{l+1}\}} y_{i_{l+1}j} = 1\\
&x_{i_{l+1}i_{l+2}} + \sum_{j\in V\setminus\{i_{l+2}\}} y_{i_{l+2}j} \leq \; x_{i_{l+1}i_{l+2}} + y_{i_{l+1}i_{l+2}} \sum_{j\in V\setminus\{i_{l+2}\}} y_{i_{l+2}j} \leq 1\\
&\vdots\\
&x_{i_{m-1}i_m} + \sum_{j\in V\setminus\{i_m\}} y_{i_mj} \leq \; x_{i_{m-1}i_m} + y_{i_{m-1}i_m} \sum_{j\in V\setminus\{i_m\}} y_{i_mj} \leq 1
\end{aligned}$$

Note that we obtain the expressions on the left hand side by omitting some of the terms of the left hand sides of inequalities (3.10). Summing them we obtain $x(P_I)+\sum_{i\in I\setminus\{i_0\}}\sum_{(i,j)\in A} y_{ij}+\sum_{j\in I:(i_0,j)\in A} y_{i_0j} \leq m$ which is the star-path inequality induced by I, showing that the inequality is not violated. $\square$

Theorem 4.4 *There is a star-path inequality (3.20) violated by $(\overline{x}, \overline{y})$ in G if and only if there is a star-path inequality violated by $(\overline{x}', \overline{y}')$ in G' .*

Proof Let $I = \{i_0, \ldots, i_m\}$ be an ordered set inducing a star path inequality violated by $(\overline{x}, \overline{y})$. First, it is clear that if $(\overline{x}', \overline{y}')$ is obtained by either θ_1 or θ_2, then I still induces a star path inequality violated by $(\overline{x}', \overline{y}')$ in G'. Also by Lemma 4.3, if $\overline{x}_e = 1$, then $e \notin P_I$. Thus θ_4 and θ_5 do not affect I and hence the star path inequality induced by I remains violated by $(\overline{x}', \overline{y}')$ in G'. Now, suppose $\overline{y}_{uv} = 1$ for some $(u, v) \in A$. Without loss of generality, we may assume that $u = i_l \in I$. Therefore, $\overline{y}'_{i_lk} = 0$ for all $k \in V \setminus \{i_l, v\}$, and $\overline{x}'_{i_lk} = 0$ for all $k \in V \setminus \{i_l\}$. As the star-path inequality induced by I is violated, $z = \overline{x}(P_I) + \sum_{i\in I\setminus\{i_0\}}\sum_{(i,j)\in A} \overline{y}_{ij} + \sum_{j\in I:(i_0,j)\in A} \overline{y}_{i_0j} > m$. Let $I' = I \setminus \{i_l\}$. To obtain the star-path inequality induced by I' from the one induced by I, we need to remove the terms related to node i_l from the left hand side and add $x_{i_{l-1}i_{l+1}}$.

Decreasing the right hand side by one we obtain the new star-path inequality induced by I'. Since $z - 1 + \overline{x}_{i_{l-1},i_{l+1}} > m - 1$, this new star-path inequality is also violated by $(\overline{x}', \overline{y}')$. So there is a violated star-path inequality in G' after reduction by θ_3, if there is one in G.

The converse of the theorem can be easily seen to be true. $\square$

These results allow us to apply the reduction operations and then solve the separation problems on the reduced spaces. Since it will be possible to identify the violated valid inequalities on the reduced spaces provided that there is a violated one in the original space, we expect to decrease the amount of time spent to solve the separation problems as the problem sizes are reduced by the reduction operations.

4.2 Separation Algorithms

In this section we shall describe our separation algorithms. For a given fractional solution (x^*, y^*) we define the following sets $V_h = \{i \in V : \sum_{j \in V \setminus \{i\}} y^*_{ij} = 0\} \cup \{0\}, V_{ph} = \{i \in V : \sum_{j \in V \setminus \{i\}} y^*_{ij} > 0 \text{ and } x^*(\delta(i)) > 0\}, V_u = \{i \in V : x^*(\delta(i)) = 0 \text{ and } \exists j \in V \setminus \{i\} \text{ such that } y^*_{ij} = 1\}$ and $V_{pu} = \{i \in V \setminus V_u : x^*(\delta(i)) = 0\}$. We call the elements of these sets *hubs*, *partial hubs*, *users*, and *partial users*, respectively. Note that V_h, V_{ph}, V_u, V_{pu} form a partition of V. We also define $V^* = V_h \cup V_{ph}$, $E^* = \{\{i,j\} \in E : x^*_{ij} > 0\}$ and $A^* = \{(i,j) \in A : 0 < y^*_{ij} < 1\}$. $G^* = (V^*, E^*)$ is our support graph and it may be disconnected. Let $G^i = (V^i, E^i)$ for $i = 0, \ldots, r$ be the i^{th} connected component of G^*. Without loss of generality, we assume $0 \in V^0$. Clearly $G^0 = G^*$ if G^* is connected. Violated clique inequalities (3.10) are found by complete enumeration and for the cut (3.12), star-path (3.20) and extended F-partition inequalities (3.18) the separation algorithms are described in the following sections.

It is important to note that all the separation procedures described are performed on the graphs obtained by the reduction operations unless otherwise specified. Then we will use some lifting procedures to obtain the violated inequalities that will be added to the model from the ones found by the separation algorithms. First we describe the lifting procedures. Let $\overline{W}$ ($\overline{V}$) be the set of nodes that arises from θ_4 (θ_5) and $|\overline{W}| = q$ ($|\overline{V}| = r$). Let W_i be the contracted node set and w_i the corresponding node for $i = 1, \ldots, q$. Similarly, let $u_i v_i$ be the contracted edge and $\overline{v}_i$ be the corresponding node for $i = 1, \ldots, r$.

Let (S', i') be a pair inducing a violated cut inequality (3.12) in G'. Without loss of generality, say $w_1, \ldots, w_j \in S'$ for some $j \in \{1, \ldots, q\}$ and $\overline{v}_1, \ldots, \overline{v}_k \in S'$ for some $k \in \{1, \ldots, r\}$. Let $S = S' \setminus (\cup_{l=1}^{j} w_l \cup_{l=1}^{k} \overline{v}_l) \cup (\cup_{l=1}^{j} W_l \cup_{l=1}^{k} \overline{V}_l)$ and

$$i = \begin{cases} i' & \text{if } i' \notin \overline{W} \cup \overline{V}, \\ v_l & \text{if } i' = \overline{v}_l \text{ for some } l = 1, \ldots, k, \\ \text{any node in } W_l & \text{if } i' = w_l \text{ for some } l = 1, \ldots, j. \end{cases}$$

Then (S, i) induce a violated cut inequality in G.

For extended F-partition inequalities (3.18) we define Y to be the set of nodes deleted by θ_3. Let $(V'_0, \ldots, V'_p)$, F' and $i'_1, \ldots, i'_p$ define a violated extended F-partition inequality in G'. Set $V_i = V'_i \setminus (\cup_{1 \le l \le q : w_l \in V'_i} \{w_l\} \cup_{1 \le l \le r : \overline{v}_l \in V'_i} \{\overline{v}_l\}) \cup (\cup_{1 \le l \le q : w_l \in V'_i} W_l \cup_{1 \le l \le r : \overline{v}_l \in V'_i} \overline{V}_l)$ for $i = 1, \ldots, p$ and $V_0 = V'_0 \setminus (\cup_{1 \le l \le q : w_l \in V'_0} \{w_l\} \cup_{1 \le l \le r : \overline{v}_l \in V'_0} \{\overline{v}_l\}) \cup (\cup_{1 \le l \le q : w_l \in V'_0} W_l \cup_{1 \le l \le r : \overline{v}_l \in V'_0} \overline{V}_l) \cup Y$. The fixed nodes should also be updated as follows:

$$i_s = \begin{cases} i'_s & \text{if } i'_s \notin \overline{W} \cup \overline{V}, \\ v_l & \text{if } i'_s = \overline{v}_l \text{ for some } l = 1, \ldots, k, \quad \text{for } s = 1, \ldots, p \\ \text{any node in } W_l & \text{if } i'_s = w_l \text{ for some } l = 1, \ldots, j. \end{cases}$$

Then $(V_0, \ldots, V_p)$, F' and $i_1, \ldots, i_p$ induce a violated extended F-partition inequality (3.18) in G.

For the star-path inequalities as the reduction operations do not affect the ordered set of nodes defining the violated star-path inequality, the ordering can also be used in G.

Having described the lifting tool, we can focus on the separation algorithms for each class of valid inequalities.

4.2.1 Cut inequalities

A cut inequality (3.12) is defined by a node set $S \subseteq V \setminus \{0\}$ and a fixed node $i \in S$. For a given $i \in V \setminus \{0\}$, it is possible to check if there exists a subset $S \subseteq V \setminus \{0\}$ with $i \in S$ for which the cut inequality is violated by solving a minimum cut problem. Let $G_i^* = (V^* \cup \{i\}, E_i^*)$ where $E_i^* = E^* \cup \{\{i,j\} : j \in V^* \text{ and } (i,j) \in A^*\}$. Set the capacity of edge e to x_e^* if $i \notin e$, and to $x_{ij}^* + 2y_{ij}^*$ otherwise. Labbé et al. [30] showed that a maximum violated cut inequality with fixed node i can be found by solving a minimum cut problem on this graph separating nodes i and 0. If the minimum cut capacity is less than 2, then there is a violated cut inequality. Therefore cut inequalities can be separated exactly by solving $|V| - 1$ minimum cut problems. We also use the separation method of [30] together with a heuristic algorithm to speed up the separation.

Our separation algorithm works in three phases. We first use the connected components of the support graph to generate violated cut inequalities. For a given connected component $G^i = (V^i, E^i)$ we compute the violation of the cut inequality defined by every $j \in V^i$ and node set $V^i \cup \{k \in V_u \cup V_{ph} : y_{jk}^* > 0\}$. As $x^*(\delta(V^i)) = 0$, it is very likely that we find a violated inequality this way. The most violated cut inequality is selected for $i = 1, \ldots, r$. If $r > 1$ we perform the same operations for $\cup_{i=1}^q V^i$ for $q = 2, \ldots, r$.

Second, we use our heuristic on G^0. The heuristic is based on the algorithm of Hao and Orlin [22] which finds a global minimum cut, i.e., a cut with the minimum capacity among all cuts of a graph. In this algorithm, $n - 1$ minimum cut problems, where n is the number of nodes of the graph, are solved. Let s_i be the source node of the i^{th} minimum cut problem and t the sink node at the beginning. In the i^{th} step, the minimum cut between s_i and $\{t, s_1, \ldots, s_{i-1}\}$ is found. Hao and Orlin [22] show that the cut with the minimum capacity among the ones found in the algorithm is the global minimum cut of the graph. Moreover, they select the source nodes in such a way that the running time of the algorithm is equivalent to that of a single minimum cut problem. We apply their algorithm on G^0 with the capacity of each edge $e \in E^0$ being equal to x_e^*. Our root node is the initial sink for the algorithm. Using this we obtain $|V^0| - 1$ cut sets, say $S_1, \ldots, S_{|V^0|-1}$. Since y^* values are ignored in edge capacities, there are three possible outcomes for every cutset. If the capacity of the cut is greater than or equal to 2 then there is no violated cut inequality associated with this cut. If the capacity is less than 2, then the violation must be calculated by taking y^* values into account for a given fixed node to see if the corresponding cut inequality is really violated. Therefore, for $i = 1, \ldots, |V^0| - 1$ such that the capacity of $[S_i, V^0 \setminus S_i]$ is less than 2, we calculate the violation of the cut inequality defined by S_i and j for every $j \in S_i \cup V_u \cup V_{ph}$. If there is at least one violated cut inequality, we choose the one with the maximum violation. If there is more than one fixed node with the same violation, then we choose the one which is used less often in previous cut inequalities. We can keep track of this information using an array. If there is still a tie, we break it arbitrarily. Note that if the fixed node, say $j \in V_u \cup V_{pu}$, is chosen for a given cutset S, the cutset must be extended as $S \cup \{j\}$ since $j \notin S$.

There are two advantages of this method. First, it is very fast. Second, if the minimum cut capacities are all greater than or equal to 2, then we can conclude that there is no violated cut inequality. Notice that we are not interested in the nodes of $V_u \cup V_{pu}$ in minimum cut computations as there is no adjacent edge to them in the support graph. These nodes are put together with node 0. But if the fixed node of a given cut inequality is assigned to a user or partial user node j then we move it from $V \setminus S$ to S. This operation does not affect the first term of the cut inequality but reduces the second term and hence the violation increases.

As finding a minimum cut between i and 0 provides the most violated cut inequality with fixed node i, we need to solve minimum cut problems for every node of the backbone network except for the root node. This is necessary as the cut inequalities are in the model formulation and must be separated exactly. However, as we use a heuristic step first, we can eliminate some nodes from consideration in the exact separation phase. This is possible because either some violated cut inequality is found for the fixed node in the heuristic phase or from the information obtained from the first step we expect that there is no violated cut inequality with this fixed node. So we define C to be the set of nodes i in $V_{pu} \cup V_{ph}$ such that i is not used as a fixed node in the inequalities added by the heuristic. The nodes of C will be referred to as candidate nodes and are the nodes which are not eliminated from consideration after the heuristic phase. We remark that nodes in $V_h \cup V_u$ are excluded from this candidate list due to the following reasons. Let $i \in V_h$ and $S \subseteq V \setminus \{0\}$ be such that $i \in S$. If $x^*(\delta(S)) < 2$, then there is a violated cut inequality and i gives the maximum violation as $\sum_{j \in V \setminus S} y^*_{ij} = 0$. So if a violated cut inequality with fixed node i is not found in the heuristic phase we can say that either there is no violated cut inequality with fixed node i or another inequality with the same cutset but with a different fixed node is found. Therefore we do not include hubs in C. A similar reasoning can be done for the users. Besides, there is no guarantee we will find a different cut for this node by using the exact separation if we find a violated cut inequality for some i in the heuristic phase. So we do not include such nodes in C, either.

If at least one cut with capacity less than 2 is found in the heuristic phase and $C \neq \emptyset$ we pass to the last step, the exact separation phase. In this phase we solve a minimum cut problem between i and 0 on G^*_i for every $i \in C$.

The Goldberg-Tarjan algorithm [17] is used to solve the minimum cut problems. Our separation procedure is given in Algorithm 1. As it turns out during our experimentation, the heuristic part significantly improves the CPU time of our branch-and-cut algorithm.

4.2.2 Extended F-partition inequalities

Unlike the cut inequalities, the separation of the extended F-partition inequalities is not easy. We claim that the separation problem associated with the extended F-partition inequalities (3.18) is NP-Hard. Here we consider the problem of finding a most violated inequality. So we define the decision version of the separation

Algorithm 1: Cut Inequality Separation

```
Input: $(x^*, y^*), G^i = (V^i, E^i)$ for $i = 0, \dots, r$, $G_i^* = (V^0 \cup \{i\}, E_i^*)$ for
       $i \in V_{pu} \cup (V_{ph} \cap V^0)$
1  begin
2      $C \leftarrow V_u \cup (V_{ph} \cap V^0)$
3      if $r > 0$ then
4          for $i = 1$ to $r$ do
5              forall $k \in V^i \cup V_u \cup V_{pu}$ do $z_k = 2 - 2\sum_{j \in V^* \setminus V^i} y_{kj}^*$
6              $j \leftarrow argmax_{k \in V^i \cup V_u \cup V_{pu}}\{z_k\}$
7              if $z_j > 0$ then
8                  $V^i \cup \{k \in V_u \cup V_{pu} : y_{jk}^* > 0\}$ and $j$ form a violated cut ineq.
9                  $C \leftarrow C \setminus \{j\}$
10         if $r > 1$ then
11             for $i = 2$ to $r$ do
12                 forall $k \in \cup_{j=1}^{i} V^j \cup V_u \cup V_{pu}$ do $z_k = 2 - 2\sum_{j \in V^* \setminus \cup_{l=1}^{i} V^l} y_{kj}^*$
13                 $l \leftarrow argmax_{k \in \cup_{j=1}^{i} V^j \cup V_u \cup V_{pu}}\{z_k\}$
14                 if $z_l > 0$ then
15                     $\cup_{j=1}^{i} V^j \cup \{k \in V_u \cup V_{pu} : y_{lk}^* > 0\}$ and $l$ form a violated
                       cut ineq.
16                     $C \leftarrow C \setminus \{l\}$
17     Use Hao-Orlin algorithm on $G^0$ and find $|V^0| - 1$ cutsets denoted by
       $S_1, \dots, S_{|V^0|-1}$
18     for $l = 1$ to $|V^0| - 1$ do
19         if capacity of $[S_l, V^0 \setminus S_l]$ is less than 2 then
20             forall $i \in S^l \cup V_u \cup V_{pu}$ do $z_i = 2 - x^*(\delta(S^l)) - 2\sum_{j \in V^* \setminus S^l} y_{ij}^*$
21             $k \leftarrow argmax_{i \in S^l \cup V_u \cup V_{pu}}\{z_i\}$
22             if $z_k > 0$ then
23                 $S \cup \{j \in V_u \cup V_{pu} : y_{kj}^* > 0\}$ and $k$ form a violated cut
                   inequality
24                 $C \leftarrow C \setminus \{k\}$
25     forall $i \in C$ do
26         Find minimum cut $[S, V^* \setminus S]$ between $i$ and 0 on $G_i^*$
27         if capacity of $[S, V^0 \setminus S]$ is less than 2 then
           $S \cup \{j \in V_u \cup V_{pu} : y_{uj}^* > 0\}$ and $i$ form a violated cut inequality
28 end
```

problem as follows.

Given a graph $G = (V, E)$, a special node $0 \in V$, a solution $(\overline{x}, \overline{y}) \in R_+^{|E|} \times R_+^{|V|^2}$, and a positive number κ, does there exist a partition $V_0, V_1, \ldots, V_p$ of V with $0 \in V_0$, $F \subseteq \delta(V_0)$ with $|F| = 2k + 1$ for some integer $k \geq 0$, $i_l \in V_l$ for $l = 1, \ldots, p$ such that $\overline{x}(\delta(V_0, \ldots, V_p) \setminus F) + \sum_{l=1}^{p} \sum_{j \in V \setminus V_l} \overline{y}_{i_l j} \leq p - k - \kappa$?

To establish the complexity status of the separation problem associated with the extended F-partition inequalities, we will use a reduction from the decision version of the *Uncapacitated Concentrator Location Problem (decUCL)*. The *decUCL* is defined as follows. Given a set of nodes I, cost C_{ij} for $i \in I$ and $j \in I$ and a positive scalar K, does there exist a nonempty subset I' of I and a choice $j_i \in I'$ for each $i \in I \setminus I'$ such that $\sum_{j \in I'} C_{jj} + \sum_{i \in I \setminus I'} C_{ij_i} \leq K$? This problem is NP-complete [31]. To avoid trivial cases, we consider instances with $\max_{i,j \in I} C_{ij} > \frac{K}{|I|}$.

Theorem 4.5 *The decision version of the separation problem associated with the extended F-partition inequalities (3.18) is NP-complete.*

Proof It is easy to verify that the problem is in NP.

To show that the problem is NP-complete, we give a polynomial time reduction of the *decUCL* to the decision version of the separation problem. Given an instance of the *decUCL*, consider the following instance of the separation problem. Let $\kappa = \max_{i,j \in I} C_{ij}|I| - K$. Set $V = \{0\} \cup I$, $E = \{\{i, j\} : i \in V, j \in V \setminus \{i\}\}$, $\overline{x} = 0$, $\overline{y}_{ij} = \frac{K+\kappa}{|I|} - C_{ji}$ for $i \in I$ and $j \in I$. Observe that $\overline{y}_{ij}$ is nonnegative for each $i \in I$ and $j \in I$. Let $\overline{y}_{i0} = \overline{y}_{0i} = 0$ for all $i \in I$ and $\overline{y}_{00} = 1$.

Notice that as $\overline{x} = 0$ and $\sum_{j \in V \setminus V_l} y_{i_l j} = 1 - \sum_{j \in V_l} y_{i_l j}$ for i_l for $l = 1, \ldots, p$, the extended F-partition inequality (3.18) can be rewritten as

$$\sum_{l=1}^{p} \sum_{j \in V_l} y_{i_l j} \leq k. \tag{4.2}$$

Let $V_0, V_1, \ldots, V_p$ be a partition of V with $0 \in V_0$, $F \subseteq \delta(V_0)$ with $|F| = 2k+1$ for some integer $k \geq 0$, $i_l \in V_l$ for $l = 1, \ldots, p$ such that the corresponding extended F-partition inequality (4.2) is violated with violation at least as large as κ. Now consider a new partition such that $V_l^{'} = V_l$ for $l = 2, \ldots, p$, $V_0^{'} = \{0\}$, and $V_1^{'} = V_1 \cup V_0 \setminus \{0\}$. Let $F^{'}$ be a subset of $\delta(0)$ with cardinality 1. The resulting extended F-partition inequality (4.2) is also violated with violation at least as large as κ. So there exists an extended F-partition inequality with violation at least κ if and only if there exists an extended F-partition inequality with violation at least κ, $V_0 = \{0\}$, and $F \subseteq \delta(0)$ with $|F| = 1$.

We claim that there exists a solution to the *decUCL* if and only if there exists a partition of V into $V_0, V_1, \ldots, V_p$ with $V_0 = \{0\}$ and $F \subseteq \delta(0)$ with $|F| = 1$, and a choice of nodes $i_1, \ldots, i_p$ with $i_l \in V_l$ for $l = 1, \ldots, p$ such that the inequality (4.2) is violated with a violation of at least κ.

Given a solution of *decUCL*, let $p = |I^{'}|$ and so $I^{'} = \{i_1, \ldots, i_p\}$. For $l = 1, \ldots, p$, let $V_l = \{i \in I \setminus I^{'} : j_i = i_l\} \cup \{i_l\}$, $V_0 = \{0\}$ and F be any element of $\delta(0)$. The left hand side of the inequality, $\sum_{l=1}^{p} \sum_{j \in V_l} \overline{y}_{i_l j}$, is equal to $\sum_{l=1}^{p} \overline{y}_{i_l i_l} + \sum_{l=1}^{p} \sum_{j \in V_l \setminus \{i_l\}} \overline{y}_{i_l j} = \sum_{i \in I^{'}} (\frac{K+\kappa}{|I|} - C_{ii}) + \sum_{i \in I \setminus I^{'}} (\frac{K+\kappa}{|I|} + C_{ij_i}) = K + \kappa - (\sum_{i \in I^{'}} C_{ii} + \sum_{i \in I \setminus I^{'}} C_{ij_i})$. As $\sum_{i \in I^{'}} C_{ii} + \sum_{i \in I \setminus I^{'}} C_{ij_i} \leq K$, we have $\sum_{l=1}^{p} \sum_{j \in V_l} \overline{y}_{i_l j} \geq \kappa$ and hence the inequality (4.2) is violated with a violation of at least κ.

Given a partition of V into $V_0, V_1, \ldots, V_p$ with $V_0 = \{0\}$, $F \subseteq \delta(0)$ with $|F| = 1$, and a choice of nodes $i_1, \ldots, i_p$ with $i_l \in V_l$ for $l = 1, \ldots, p$ such that the inequality (4.2) is violated with a violation of at least κ, let $I^{'} = \{i_1, \ldots, i_p\}$, and $j_k = i_l$ for $k \in V_l$ and $l \in \{1, \ldots, p\}$. We can show that this is a solution to the *decUCL* following the steps above. $\square$

For this reason, we propose two heuristic algorithms for the separation of the extended F-partition (3.18) inequalities instead of an exact separation method. The first one is based on searching odd fractional cycles in G^*. A set of nodes $\{v_1, \ldots, v_p\}$ which induces an odd cycle is determined, if one exists. Let $V_0 = V \setminus \{v_1, \ldots, v_p\}$. The edges in $\delta(V_0)$ with values greater than $\frac{1}{2}$ are included in F in such a way that $|F|$ becomes odd. The corresponding inequality is checked for violation.

The second heuristic is based on finding a maximum cut in a graph. Since the maximum cut problem is NP-hard, it is solved heuristically as follows. We associate with each edge e a capacity equal to $1-x^*_e$. We use the algorithm of Hao and Orlin [22] on G^0 to find $|V^0|-1$ minimum cuts. Let S be a cutset obtained by this algorithm such that $0 \in S$ and $V_0 = S \cup V_u \cup V_{pu}$. Let $V^0 \setminus S = \{v_1, \ldots, v_p\}$. Then our partition is $(V_0, V^1, \ldots, V^r, v_1, \ldots, v_p)$. We construct F as we did in the first heuristic. For the strong component G^i, node $j \in V^i$ with the largest $\sum_{k \in V \setminus V^i} y^*_{jk}$ value is selected as the fixed node for $i = 1, \ldots, r$. We check the extended F-partition inequality induced by this partition and fixed nodes to see if it is violated.

Note that we included the connected components as separate node sets while forming the partition. This is because doing so increases the violation of a given extended F-partition inequality. Consider a connected component G^i. Since $x^*(\delta(V^i)) = 0$ and $\sum_{k \in V \setminus V^i} y^*_{jk} \leq 1$ for any fixed node $j \in V^i$, the increase in the left hand side of the inequality is at most 1. But the right hand side increases exactly by one when we include the connected component in the partition. Therefore using a connected component, we can increase the violation of a given extended F-partition inequality.

4.2.3 Star-path inequalities

The separation problem associated with the star-path inequalities (3.20) is also NP-Complete. First we remark that inequality (3.20) can be rewritten as $x(P_I) - \sum_{l=1}^m y_{i_l i_l} + \sum_{l=1}^m y_{i_0 i_l} \leq 0$ using the self assignment variables.

The decision version of the separation problem is then defined as follows. Given a graph $G = (V, E)$, $K > 0$, a special node $0 \in V$, a solution $(\overline{x}, \overline{y}) \in R_+^{|E|} \times R_+^{|V|^2}$, does there exist a set of $m+1$ distinct nodes $i_0, i_1, \ldots, i_m$ in $V \setminus \{0\}$ such that $\overline{x}(P_I) - \sum_{l=1}^m \overline{y}_{i_l i_l} + \sum_{l=1}^m \overline{y}_{i_0 i_l} \geq K$?

Theorem 4.6 *The decision version of the separation problem associated with the star-path inequalities (3.20) is NP-complete.*

Algorithm 2: Extended F-Partition Inequality Separation

```
   Input: $(x^*, y^*), G^i = (V^i, E^i)$ for $i = 0, \dots, r$
1  begin
2      repeat
3          Find a fractional odd cycle $v_1, \dots, v_p$ such that $v_i \in V^0 \setminus \{0\}$ for
           $i = 1, \dots, p$
4          $V_0 \leftarrow V \setminus \{v_1, \dots, v_p\}$
5          Construct $F \subseteq \{e \in \delta(V_0) : x^*_e > 0.5\}$ so that $|F|$ is odd
6          Compute the violation for $F$ and the partition $(V_0, v_1, \dots, v_p)$
7      until no fractional odd cycle is found ;
8      if no violated extended $F$-partition inequality is found above then
9          Use algorithm of Hao and Orlin on $G^0$
10         foreach cut $[S, V^0 \setminus S]$ such that $0 \in S$ found in the algorithm do
11             $V_0 \leftarrow S \cup V_u \cup V_{pu}$
12             Construct $F \subseteq \{e \in \delta(V_0) : x^*_e > 0.5\}$ so that $|F|$ is odd
13             Compute the violation for $F$ and the partition
               $(V_0, V^1, \dots, V^r, v_1, \dots, v_p)$ where $V^0 \setminus S = \{v_1, \dots, v_p\}$
14 end
```

Proof NP membership is easily verifiable. To establish the complexity status we give a reduction from the *Hamiltonian Path Problem* which is defined as follows. Given a graph $G' = (V', E')$, does G' contain a simple path consisting of all the nodes in V'? Given such an instance, consider the following instance of the separation problem. Set $V = V' \cup \{0\}$, $E = \{\{i, j\} : i \in V, j \in V \setminus \{i\}\}$ $\overline{x}_e = 1$ for $e \in E'$, $\overline{x}_e = 0$ for $e \in E \setminus E'$, $\overline{y} = 0$, and $K = |V| - 1$. Now it is easy to conclude that G' has a Hamiltonian Path if and only if the constructed separation problem has a solution. □

This result makes the use of a heuristic algorithm for separation, necessary.

Assuming that all clique inequalities (3.10) are satisfied, it can be seen that a star-path inequality can be violated only if the x terms of the inequality are positive. Moreover, we observed that violated star-path inequalities are frequently induced by paths which include nodes to which the initial node s is assigned. For this reason we restrict our search for paths on these nodes. For some initial node s, let $V_s = \{i \in V^* \setminus \{0\} : y^*_{si} > 0\} \cup \{s\}$ and $E_s = \{ij : i \in V_s, j \in V_s, x^*_{ij} < 1\}$.

Note that we omitted the edges with values 1 by Lemma 4.3. The edges of E_s are ordered in a decreasing way and edges are selected from this list until a simple path starting at node s and covering all nodes of V_s is obtained. After finding such a path we analyze the nodes to see if their removal increases the amount of violation of the inequality. We restrict the heuristic to the search of star-path inequalities with more than 4 nodes as the 3 and 4 node star-path inequalities are generated by enumeration. Using the necessary conditions stated in Lemma 4.3 for a star-path inequality to be violated, the enumeration can be performed very efficiently.

Algorithm 3: Star-Path Inequality

```
   Data: $(x^*, y^*), S = \{i \in V : \exists j | 0 < y^*_{ij} < 1\}$
1  begin
2      forall $s \in S$ do
3          $V_s = \{i \in V^* \setminus \{0\} : y^*_{si} > 0\} \cup \{s\}$
4          $E_s = \{ij : i \in V_s, j \in V_s, x^*_{ij} < 1\}$
5          $G_s = (V_s, E_s)$
6          if $|V_s| > 4$ then
7              Find the longest path starting from $s$ on $G_s$ by a greedy method
8              Let $P = v_0, \ldots, v_p$ be the path
               $violation = \sum_{i=0}^{p-1} x^*_{v_i v_{i+1}} + \sum_{i=1}^{p} y^*_{v_0 v_i} + \sum_{i=1}^{p} \sum_{j \in V \setminus \{v_i\}} y^*_{v_i j} - p$
9              repeat
10                 for $i = 1$ to $p$ do
                   $z_i = 1 + x^*_{v_{i-1} v_{i+1}} - x^*_{v_i v_{i+1}} - x^*_{v_{i-1} v_i} - y^*_{v_0 v_i} - \sum_{j \in V \setminus \{v_i\}} y^*_{v_i j}$
11                 $k \leftarrow argmax_{i=1,\ldots,p}\{z_i\}$
12                 $\delta \leftarrow z_k$
13                 if $\delta >= 0$ then
14                     $violation \leftarrow violation + \delta$
15                     Remove $v_k$ from $P$
16             until $\delta < 0$ or $|P| \leq 4$ ;
17             if $violation > 0$ then $P$ induces a violated star-path inequality
18     Find violated 3 and 4 node star-path inequalities by enumeration
19 end
```

4.3 Computational results

Based on our polyhedral analysis and the separation algorithms described earlier we developed a branch-and-cut algorithm for the 2ECSSP. We start the optimization by solving the following linear program:

$$\begin{aligned}
&\min \sum_{ij\in E} c_{ij}x_{ij} + \sum_{(i,j)\in A} d'_{ij}y_{ij} && \\
&\text{s.t.} && \\
&\quad x_{0i} + \sum_{j\in V\setminus\{i\}} y_{ij} \le 1 && \forall i \in V\setminus\{0\} \\
&\quad x(\delta(i)) + \sum_{j\in V\setminus\{i\}} y_{ij} \ge 2 && \forall i \in V\setminus\{0\} \\
&\quad 0 \le x_{ij} \le 1 && \forall ij \in E \\
&\quad 0 \le y_{ij} \le 1 && \forall (i,j) \in A
\end{aligned}$$

The solution $(\overline{x},\overline{y})$ of this initial subproblem is feasible for 2ECSSP if it satisfies the cut inequalities, the clique inequalities and integrality constraints. Therefore at each iteration of the branch-and-cut algorithm we solve the separation problems to determine if there are violated inequalities. The different inequalities are separated in the following order: clique (3.10), cut (3.12), F-partition (3.18) and star-path (3.20) inequalities. We generate up to 200 violated valid inequalities at each iteration and look for a violated inequality only if we cannot find a violated inequality of a previous class and found less than 200 inequalities. Note that if a solution is integral and there is no violated cut or clique inequality, then we do not need to solve the separation problems for the F-partition and the star-path inequalities.

We need to note that there are many factors that could affect the overall performance of the branch-and-cut algorithm and the number of violated inequalities that are added at each iteration is one of these factors. If too many violated inequalities are added at each iteration, then the size of the subproblems and the cpu time required to solve them will increase rapidly. On the other hand, if we add too few violated inequalities then the number of iterations will increase and the total cpu time may increase, as well. Unfortunately, there is no exact way to determine how many inequalities should be added at each iteration. For this reason, we make experiments using different limits for the number of inequalities to be added. Although, different limits have different effects on different instances, meaning that cpu time for some instances decreased while it increased for the others, we observed that adding 200 violated inequalities provided better performance for the branch-and-cut algorithm. It is also possible to set this limit to a percent of the size of the problem instance instead of a constant number. Such a strategy may not have very significant effects in our experiments, since the size of the problem instances we solved do not differ very much. However, if very large problems are to be solved, then such a strategy should be considered.

We adapt the instance generation method of Labbé et al. [30]. Our test problems are based on TSP instances from TSPLIB 2.1 [38]. TSPLIB is a library that includes instances for the TSP and similar problems. The instances are obtained from different resources and they have different metrics, such as two and three dimensional Euclidean metric, geographical distances, and etc. The variety of the metrics used in the instances will allow us to observe the performance of the proposed solution methodology on different types of problems instances. Name of each problem instance is composed of a string and a number, which denotes the number of nodes in the instance. The detailed information about the instances and the methods to compute the distances between nodes are provided in [38]. Note that, these instances only define the distances between the nodes. However, we need two different cost parameters, one for the backbone links and one for the assignments, for every node pair. To compute the backbone link setup costs and assignment costs we use the following formulas proposed by Labbé et al. [30], $c_{ij} = \lceil \alpha l_{ij} \rceil$ and $d_{ij} = \lceil (10 - \alpha) l_{ij} \rceil$ where l_{ij} denotes the distance between nodes i and j in the TSPLIB instances and $\alpha \in \{3, 5, 7, 9\}$. We set $d_{ii} = 0$ for all $i \in V \setminus \{0\}$. Note that as α decreases, assignment costs increase while backbone link setup costs decrease, i.e., the problem gets closer to the 2-edge connected subgraph problem. Conversely, as α increases the number of nodes chosen to be hubs decreases and the problem gets farther from the 2-edge connected subgraph problem. The α is an artificial coefficient used to obtain instances with different cost structures although the underlying graphs are the same. The aim of this is to observe the effectiveness of the proposed solution methodology on different problem instances. In real life problems, the cost structure will be formed based on several factors, such as geographical or technological constraints. So there will be no need for an α value in real life problems. This also means that there is not a correct α value for the experiments, however, the set of possible values of α is chosen such that almost all nodes are chosen to be hubs in the optimal solution when $\alpha = 3$ and very few hubs are used in the backbone network in the optimal solution when $\alpha = 9$. In addition to these extreme cases, setting $\alpha \in \{5, 7\}$ allows us to observe the performance of the branch-and-cut algorithm on the intermediate ones.

A construction heuristic is used to find an initial solution at the beginning of the algorithm. We start from node 0 and apply the nearest neighbor TSP heuristic to obtain a cycle that includes all nodes of the graph. As this cycle is 2-edge connected, it forms a feasible solution. Throughout the branch-and-cut algorithm, we also use an LP based improvement heuristic. The edges are ranked in a decreasing order according to their values in the fractional solution. We start with an empty set and add the edges one by one until we obtain a cycle. The remaining nodes which are not included in the cycle are assigned to a node in the cycle with minimum assignment cost.

We implemented our algorithm in C++ using ABACUS 3.0 as the framework and CPLEX 11.0 as the LP solver. Computational analysis is performed on a workstation with 2.66 GHz xeon processor and 8 Gb of ram. We use best first search as the search strategy and strong branching as the branching strategy. Tailing off control is used; we branch if the improvement in the objective function value is small in 10 subsequent iterations. We do not use pool separation, and added inequalities are removed if they are not active in 5 subsequent iterations.

Our computational results are provided in Tables 4.1-4.6. In all the tables, the first two columns denote the name of the instance and the α value, respectively.

The number of nodes in the instance is also included in the name of the instance. The other columns given in the tables are explained below:

NClq number of generated clique inequalities;

NCut number of generated cut inequalities;

NFP number of generated F-partition inequalities;

NSP number of generated star-path inequalities;

Opt optimal objective function value;

NNode number of branch-and-cut nodes evaluated;

Cpu total CPU time in seconds (rounded to the nearest integer);

Gap1 the relative error between the optimal solution value and the lower bound obtained at the root node of the branch-and-cut tree;

Gap2 the relative error between the best upper bound (the optimal solution value if the problem is solved to optimality) and the lower bound obtained at the end of the branch-and-cut algorithm.

In Table 4.1 only clique and cut inequalities are used in solving the instances. In Table 4.2, we present the results obtained using extended F-partition inequalities together with the basic inequalities. The last three columns show the improvement in the gap, CPU time and the number of branch-and-cut nodes attained by adding the extended F-partition inequalities, respectively. Here a negative value means the performance gets worse after adding the extended F-partition inequalities. If an instance is not solved to optimality using only clique and cut inequalities, then it is not possible to compute the improvement for CPU time; we denote such cases with $**$. Table 4.3 includes the results obtained using both extended F-partition and star-path inequalities together with the basic inequalities, and its columns are similar to those of Table 4.2 except that the improvement columns give the comparisons between Tables 4.2 and 4.3. Finally, the instances are solved with all the valid inequalities without using the reduction operations and the results are given in Table 4.4. The last three columns of this table present information about comparisons made between Tables 4.3 and 4.4, and a positive value indicates that the algorithm performs better with reduction operations.

We solved 28 instances, 10 of which reached optimality at the root node. We observe that the number of branch-and-cut nodes exploited tends to decrease as α increases. The optimal solution is found at the root node when $\alpha = 9$ even if the F-partition and star-path inequalities are not used. However, when $\alpha = 3$ and no F-partition and star-path inequalities are used, the number of nodes increases to an extent that the algorithm terminates before reaching the optimal solution due to memory problems. When only the clique and cut inequalities are used, 8 instances could not be solved to optimality due to memory problems and these are denoted with $*$ in the CPU column. For such instances, Gap2 could be as large as 23% and the average Gap2 is around 8.75%. The average Gap1 is 0.41%. For the instances which could not be solved to optimality, Gap2 is greater than Gap1. This is because we used the optimal solution values obtained from Table 4.2 to compute Gap1 for these instances, while the best feasible solution values were used in the computation of Gap2. The details of the results can be seen in Table 4.1.

When we also use the extended F-partition inequalities, optimal solutions could be found for all instances. Gap1 decreased as well. The average Gap1 reduced to 0.22% and 2 more instances could be solved at the root node. This shows that extended F-partition inequalities are important for the success of the branch-and-cut algorithm. For a few instances, the CPU time increased slightly, but for most of the instances, especially for the instances with $\alpha \in \{3, 5\}$, improvement in solution time is significant. These results are presented in Table 4.2.

Observing Table 4.3, we can say that the performance of the algorithm gets better on average by the addition of the star-path inequalities,. Although there are a few instances on which the algorithm performs worse, the improvements are more significant. The average Gap1 reduced to 0.19%. We can see that optimal solutions are found for all of the instances in less than 21 minutes. Without

instance	α	Opt	Gap1	Gap2	NClq	NCut	Cpu	NNode
kroA150	3	79572	0.85	0	170	1131	111	1109
kroA150	5	125435	0.08	0	224	320	1	5
kroA150	7	140961	0.25	1.2	718	3652	*	2641
kroA150	9	113080	0	0	1866	4085	231	1
kroB150	3	78180	1.26	9.18	1112	5014	*	2856
kroB150	5	122875	1.19	2.28	1959	170593	*	2832
kroB150	7	135382	0	0	690	2002	10	1
kroB150	9	108885	0	0	1906	4229	227	1
u159	3	126240	0.37	0	52	116	11	69
u159	5	204250	0.54	0	306	3939	50	113
u159	7	235221	0	0	724	2993	17	1
u159	9	199552	0	0	2038	5161	281	1
rat195	3	6957	0.85	13.97	626	2047	*	1692
rat195	5	11320	0.3	0	608	6640	168	523
rat195	7	12319	0.02	0	1004	6470	113	3
rat195	9	8977	0	0	2849	8749	872	1
d198	3	47340	0.43	0	178	1237	387	1943
d198	5	76945	2.18	2.54	3970	532153	*	1629
d198	7	94300	0.19	0	1132	9739	465	209
d198	9	96088	0	0	2864	12565	1099	1
kroA200	3	87951	0.86	23.27	1052	3740	*	1606
kroA200	5	138885	0.53	6.15	1793	133500	*	1593
kroA200	7	158227	0.37	0	1256	35163	2639	185
kroA200	9	122594	0	0	2586	7170	761	1
kroB200	3	88311	0.92	11.37	616	2239	*	1607
kroB200	5	138905	0.3	0	449	13661	240	253
kroB200	7	156638	0	0	886	3210	31	1
kroB200	9	124043	0	0	2628	6675	819	1

Table 4.1: Results with only clique and cut inequalities.

instance	α	Opt	Gap1	NFP	Cpu	NNode	gapimp	cpuimp	nodeimp
kroA150	3	79572	0.54	427	31	159	36.59	72.07	85.66
kroA150	5	125435	0	58	0	1	100	100	80
kroA150	7	140961	0.24	53	70	19	2.59	**	99.28
kroA150	9	113080	0	10	266	1	0	-15.15	0
kroB150	3	78180	0.83	633	44	331	34.01	**	88.41
kroB150	5	122875	0.72	1415	207	413	39.12	**	85.42
kroB150	7	135382	0	9	10	1	0	0	0
kroB150	9	108885	0	12	211	1	0	7.05	0
u159	3	126240	0.21	71	2	15	43.44	81.82	78.26
u159	5	204250	0.16	547	51	95	69.48	-2	15.93
u159	7	235221	0	10	16	1	0	5.88	0
u159	9	199552	0	12	270	1	0	3.91	0
rat195	3	6957	0.62	1398	128	191	27.12	**	88.71
rat195	5	11320	0.14	162	35	59	52.94	79.17	88.72
rat195	7	12319	0	6	76	1	100	32.74	66.67
rat195	9	8977	0	3	938	1	0	-7.57	0
d198	3	47340	0.28	421	114	397	34.8	70.54	79.57
d198	5	76945	0.5	507	83	23	76.92	**	98.59
d198	7	94300	0.13	51	372	155	33.01	20	25.84
d198	9	96088	0	36	1095	1	0	0.36	0
kroA200	3	87951	0.59	3970	476	1217	30.82	**	24.22
kroA200	5	138885	0.22	518	194	69	57.92	**	95.67
kroA200	7	158227	0.19	118	305	35	49.41	88.44	81.08
kroA200	9	122594	0	8	748	1	0	1.71	0
kroB200	3	88311	0.28	378	48	131	69.36	**	91.85
kroB200	5	138905	0.22	469	136	73	25.12	43.33	71.15
kroB200	7	156638	0	7	39	1	0	-25.81	0
kroB200	9	124043	0	11	778	1	0	5.01	0

Table 4.2: Results with the extended F-partition inequalities.

instance	α	Opt	Gap1	NSP	Cpu	NNode	gapimp	cpuimp	nodeimp
kroA150	3	79572	0.54	0	31	159	0	0	0
kroA150	5	125435	0	0	0	1	0	0	0
kroA150	7	140961	0.24	640	48	17	0	31.43	10.53
kroA150	9	113080	0	1128	225	1	0	15.41	0
kroB150	3	78180	0.83	0	44	331	0	0	0
kroB150	5	122875	0.72	34	136	291	-0.23	34.3	29.54
kroB150	7	135382	0	515	7	1	0	30	0
kroB150	9	108885	0	1874	224	1	0	-6.16	0
u159	3	126240	0.21	0	2	15	0	0	0
u159	5	204250	0.15	6	25	31	5.95	50.98	67.37
u159	7	235221	0	482	12	1	0	25	0
u159	9	199552	0	1823	299	1	0	-10.74	0
rat195	3	6957	0.62	0	128	191	0	0	0
rat195	5	11320	0.14	2	35	59	0	0	0
rat195	7	12319	0	534	59	1	0	22.37	0
rat195	9	8977	0	181	728	1	0	22.39	0
d198	3	47340	0.28	0	114	397	0	0	0
d198	5	76945	0.08	2	29	13	84.5	65.06	43.48
d198	7	94300	0.13	1778	339	123	0	8.87	20.65
d198	9	96088	0	3197	1258	1	0	-14.89	0
kroA200	3	87951	0.59	0	476	1217	0	0	0
kroA200	5	138885	0.22	4	194	69	0	0	0
kroA200	7	158227	0.16	668	349	73	16.72	-14.43	-108.57
kroA200	9	122594	0	1023	850	1	0	-13.64	0
kroB200	3	88311	0.28	0	47	131	0	2.08	0
kroB200	5	138905	0.22	32	116	83	0	14.71	-13.7
kroB200	7	156638	0	448	38	1	0	2.56	0
kroB200	9	124043	0	556	830	1	0	-6.68	0

Table 4.3: Results with the extended F-partition and star-path inequalities.

F-partition and star-path inequalities, 651 branch-and-cut nodes are necessary to reach the optimal solution on average while only 115 nodes are required if F-partition and star-path inequalities are used.

When we compare the results obtained using the extended F-partition and the star-path inequalities together with the results obtained by only using the extended F-partition inequalities, there is a slight improvement on average in terms of the number of nodes and the solution time. We also observe that violated extended F-partition inequalities can be found for all values of α while violated star-path inequalities could not be found for $\alpha = 3$. This is consistent with our theoretical findings as most of the nodes are selected as hubs when $\alpha = 3$. The number of the star-path inequalities is also small for $\alpha = 5$. They are mainly violated when $\alpha \in \{7, 9\}$. But for such cases the optimal solution is found at the root node, so the effect of the star-path inequalities is not as much as the effect of the extended F-partition inequalities.

The results obtained by not using the reduction operations are given in Table 4.4. It is seen that the solution times improved 13% on average with reduction. In 17 instances the solution time improved while in 8 instances we have longer solution times. It can also be seen that the reduction operations are more effective for $\alpha \in \{3, 5\}$. This behavior is consistent with the structures of the fractional solutions obtained from instances with different α values. When α is small, the number of nodes chosen to be hubs increase since the assignment costs are comparatively higher than the costs of installing backbone links. This results in higher number of nodes with y_{ii} values close to 1 and thus the number of fractional y variables decreases. Similarly, the number of x variables with fractional values also decreases. Note that the reduction operations can be applied to edges and arcs with integral values. Therefore, the reduction in the number of nodes, edges, and arcs is much more significant when α is small.

In Table 4.5 we provide the results of some larger instances. From this table, it can be seen that instances with up to 318 nodes could be solved to optimality in less than 2 hours.

instance	α	Gap1	Cpu	NNode	gapimp	cpuimp	nodeimp
kroA150	3	0.53	49	297	-0.71	36.7	46.5
kroA150	5	0	1	1	0	100.0	0.0
kroA150	7	0.24	52	17	0	7.7	0.0
kroA150	9	0	230	1	0	2.2	0.0
kroB150	3	0.83	54	337	0	18.5	1.8
kroB150	5	0.72	203	343	-0.45	33.0	15.2
kroB150	7	0.01	13	3	100	46.2	66.7
kroB150	9	0	226	1	0	0.9	0.0
u159	3	0.2	9	13	-3.14	77.8	-15.4
u159	5	0.17	53	55	9.46	52.8	43.6
u159	7	0.01	30	3	100	60.0	66.7
u159	9	0	299	1	0	0.0	0.0
rat195	3	0.62	133	191	0	3.8	0.0
rat195	5	0.11	63	65	-23.08	44.4	9.2
rat195	7	0	57	1	0	-3.5	0.0
rat195	9	0	728	1	0	0.0	0.0
d198	3	0.28	133	415	0	14.3	4.3
d198	5	0.1	93	49	20	68.8	73.5
d198	7	0.15	253	67	10.52	-34.0	-83.6
d198	9	0	1255	1	0	-0.2	0.0
kroA200	3	0.59	602	1207	0	20.9	-0.8
kroA200	5	0.23	126	55	2.22	-54.0	-25.5
kroA200	7	0.17	182	19	7.09	-91.8	-284.2
kroA200	9	0	838	1	0	-1.4	0.0
kroB200	3	0.28	67	195	-2.88	29.9	32.8
kroB200	5	0.22	81	59	0	-43.2	-40.7
kroB200	7	0	31	1	0	-22.6	0.0
kroB200	9	0	830	1	0	0.0	0.0

Table 4.4: Results without reduction operations.

instance	α	Opt	Gap1	NClq	NCut	NFP	NSP	Cpu	Nnode
pr226	3	241107	0.04	70	209	131	0	5	5
pr226	5	383055	0.69	429	762	192	9	10	5
pr226	7	469493	0.01	981	4393	12	743	85	3
pr226	9	470711	0	2638	7537	17	1254	596	1
gr229	3	401445	0.28	92	597	732	0	121	259
gr229	5	622905	0.07	382	2617	189	7	44	11
gr229	7	680052	0.02	1084	4620	22	154	121	9
gr229	9	509687	0	3148	8848	11	793	1756	1
gil262	3	7116	0.2	118	370	319	0	53	111
gil262	5	11235	0.07	454	982	211	3	22	13
gil262	7	12497	0	1279	8125	14	742	139	1
gil262	9	9749	0	3958	12226	11	1794	3764	1
lin318	3	126087	0.17	182	400	277	0	75	49
lin318	5	202140	0.15	618	16769	957	24	490	87
lin318	7	229449	0	1545	9973	9	660	340	1
lin318	9	177089	0	4547	18835	7	411	6299	1

Table 4.5: Results of larger instances.

Finally, in Table 4.6, we provide the computational results in which the heuristic phase of cut inequality (3.12) separation is not used. Looking at the solution times, it can be seen that the CPU times increase significantly. The CPU time is 164% higher on average if the heuristic part is not used. Moreover, for some instances, the CPU times are about 11 times larger without the heuristic cut separation, and hence, our heuristic separation for the cut inequalities seems to be very efficient.

4.4 Conclusion

We analyzed the 2ECSSP in Chapters 3 and 4 and presented the details of the polyhedral analysis as well as the solution approach we used. We provided some valid inequalities to make the formulation stronger. We also discussed the separation problems for the constraints and the valid inequalities. Through some reduction operations we identified methods to reduce the sizes of the problems

instance	α	Cpu	NNode	cpuimp
kroA150	3	65	495	110
kroA150	5	1	1	0
kroA150	7	67	17	40
kroA150	9	254	1	13
kroB150	3	141	915	220
kroB150	5	409	363	201
kroB150	7	11	1	22
kroB150	9	231	1	3
u159	3	3	21	50
u159	5	27	5	8
u159	7	19	1	58
u159	9	306	1	2
rat195	3	823	2765	543
rat195	5	76	55	117
rat195	7	75	1	27
rat195	9	928	1	27
d198	3	1397	3235	1125
d198	5	319	73	1000
d198	7	443	43	31
d198	9	1295	1	3
kroA200	3	642	1165	35
kroA200	5	315	73	62
kroA200	7	548	53	57
kroA200	9	860	1	1
kroB200	3	425	1047	804
kroB200	5	242	65	109
kroB200	7	51	1	34
kroB200	9	884	1	7

Table 4.6: Results without heuristic cut separation.

that must be solved for separation purposes. In order to test the effectiveness of the solution approach we employed, we developed a branch-and-cut algorithm.

Problem instances up to 318 nodes could be solved using the branch-and-cut algorithm. These results showed that our branch-and-cut algorithm is an effective method in solving 2ECSSP. In addition, we observed that the effects of the valid inequalities to the performance of the solution method. The linear programming relaxation values and solution times were improved using the valid inequalities. Similarly the reduction operations were also shown to be useful for the performance of the proposed methodology.

We have noted in Section 3.3 that the polyhedral results for the 2ECSSP can be extended to the kECSSP. Similarly, the separation problems and consequently the branch-and-cut algorithm developed for the 2ECSSP can also be used to solve the kECSSP with slight modifications. The reduction operations can also be modified so that they will be valid and useful for the kECSSP, too. However, it should not be forgotten that, there may be other valid inequalities which might be very effective in the solution of the kECSSP. Ignoring them may restrict the size of the problems that could be solved using the branch-and-cut algorithm.

Our next step will be extending the survivability to the local access networks. The results obtained during the analysis of the 2ECSSP, will also be useful for the analysis of the 2ECSDHP.

Chapter 5

Hierarchical Survivable Network Design Problem with Dual Homing

The problem of designing a two level communication network with a survivable backbone is analyzed in the previous chapters. As we have stated in Chapter 1 the survivability of the access networks should also be considered. This is because in case of a failure in the access network the user is disconnected from the network which is not desired by the user as he/she cannot receive any service. Clearly, such a failure affects less number of users, particularly only a single user if the access networks are stars as studied in Chapters 3 and 4. This gives more importance to the survivability of the backbone network, however, survivable access networks would increase the overall service quality provided by the network. In this chapter, we incorporate the survivability of the access networks in the network design problem. The problem we study is a variant of 2ECSSP, in which the backbone network is 2-edge connected and the access networks are stars, i.e. every user is assigned to one hub. We stick to the 2-edge connectivity of the backbone network and modify the structure of the access networks by assigning every user to an additional hub. Therefore each user is assigned to two distinct hubs and this results in a survivable access network. This structure can be seen in the literature and is known as *dual homing*. Therefore we call this problem *2-edge connected subgraph with dual homing problem* (2ECSDHP). It can be seen that the definitions of 2ECSSP and 2ECSDHP are quite similar and we use almost the same notation used for 2ECSSP with few changes. However, we provide the formal definition and notation below, so that this chapter is a complete one.

5.1 Problem Definition and Notation

We again consider directed and undirected graphs. An undirected graph is denoted by $G = (V, E)$ where V is the node set and E is the edge set of G. If $e \in E$ is an edge between two nodes i and j, then we also write $e = ij$ or $e = \{i, j\}$ to denote e. If a node i is one of the endpoints of an edge e we say $i \in e$, and $i \notin e$, otherwise. For two node subsets V_1 and V_2, such that $V_1 \cap V_2 = \emptyset$, $[V_1, V_2]$ denotes the set of edges having one node in V_1 and the other in V_2. Given a set $S \subseteq V$, we use $\delta(S)$ to represent the cut of S, which is the edge set consisting of edges having exactly one node in S. In other words, $\delta(S) = [S, V \setminus S]$. For $i \in V$, we will write $\delta(i)$ instead of $\delta(\{i\})$.

A directed graph with the node set V and the arc set A is represented by $D = (V, A)$. Let $a \in A$ be an arc from node i to node j, then we write $a = (i, j)$ to denote a. For $S \subseteq V$, we let $G(S)$ $(D(S))$ denote the subgraph of G (D) induced by S, that is the subgraph whose node set is S and edge (arc) set is $E(S)$ $(A(S))$, the set of edges (arcs) in G (D) having both nodes in S. Given a vector $x \in \mathbb{R}^{|E|}$ and $F \subseteq E$, we let $x(F) = \sum_{e \in F} x_e$.

For the 2ECSDHP, we assume that $V = \{0, 1, \ldots, n\}$ is a given set of terminals. We assume node 0 is a special one called *root node* and it is assumed that it corresponds to a concentrator in the backbone. As in the 2ECSSP, the root node is always a hub. Set of undirected edges $E = \{\{i, j\} : i \in V, j \in V \setminus \{i\}\}$ represents the set of potential backbone links and there is a nonnegative fixed setup cost c_e associated with every backbone link $e \in E$. We assume a complete graph in terms of edges and do not allow multiple edges. We also define a set of directed arcs $A = \{(i, j) : i \in V, j \in V\}$, which will be used to denote the assignments of users to hubs. Similarly, there is a nonnegative assignment cost d_{ij} associated with assigning terminal $i \in V$ to concentrator $j \in V$. In addition, there is the cost of d_{ii} associated with every node $i \in V$ which is the cost of installing a concentrator at node $i \in V$.

In 2ECSDHP, for a given V, the objective is to find a partition of V into C and T such that $0 \in C$. Based on this partition a set of backbone links $E' \subseteq E(C)$ such that the graph (C, E') is two edge connected is found and an assignment of each node in T to two nodes in C such that the total cost of installing backbone links and concentrators and assigning terminals to concentrators is minimum. Basically the difference between 2ECSSP and 2ECSDHP is that we assign each user to two hubs instead of one hub.

5.2 Mathematical Models

Due to the reasons that will be explained in the next section, we develop three mathematical models for 2ECSDHP. For our first model, we define the following decision variables.

$$x_e = \begin{cases} 1, & \text{if } e \text{ is used in the backbone network} \\ 0, & \text{otherwise} \end{cases}$$

$$y_{ij} = \begin{cases} 1, & \text{if } i \text{ is assigned to node } j \\ 0, & \text{otherwise} \end{cases}$$

$$t_i = \begin{cases} 1, & \text{if } i \text{ is a hub} \\ 0, & \text{otherwise} \end{cases}$$

Note that the x and y are the variables used for the 2ECSSP. To identify whether a given node $i \in V$ is a hub or not, we used variable t_i instead of y_{ii} to be consistent with the other models to be presented in this section. Using these binary variables, we formulate the following model for 2ECSDHP. This model will be referred to as M1.

$$\min \sum_{e\in E} c_e x_e + \sum_{(i,j)\in A} d_{ij} y_{ij} + \sum_{i\in V\setminus\{0\}} d_{ii} t_i \tag{5.1}$$

s.t.

$$2t_i + \sum_{j\in V\setminus\{i\}} y_{ij} = 2 \qquad \forall i \in V \setminus \{0\} \tag{5.2}$$

$$x_{ij} + y_{ij} \leq t_j \qquad \forall (i,j) \in A \tag{5.3}$$

$$x(\delta(S)) \geq 2t_i + \sum_{j\in S\setminus\{i\}} y_{ij} \qquad \forall S \subseteq V \setminus \{0\}, \forall i \in S \tag{5.4}$$

$$x_e \in \{0,1\} \qquad \forall e \in E$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in A:\ i \neq j$$

$$t_i \in \{0,1\} \qquad \forall i \in V \setminus \{0\}$$

M1 is similar to the formulation proposed for the 2ECSSP. The objective function (5.1) minimizes the cost of backbone and local access networks. Constraint (5.2) is the assignment constraint stating that a node is either made a hub or is assigned to two hubs. Constraint (5.3) is the clique constraint of the 2ECSSP. Note that we used the term *clique inequality* in Chapters 3 and 4 since the variables in this inequality were forming a clique in the conflict graph for 2ECSSP. However, it is not appropriate to use the same term here since they do not form a clique. As Constraints (5.3) define the relations between the edges of the backbone network, the assignments of the users to hubs and the type of the nodes, we will use the term *relation inequality* to refer to these constraints. Also constraint (5.4) is similar to the cut constraint of 2ECSSP and is used to make the backbone network 2-edge connected. However, the coefficients of the y variables on the left hand side are 1 instead of 2. This is because, the fixed node i of the constraint may be assigned to two nodes in S. If i is assigned to one node from S and one node from $V \setminus S$ at the same time, the left hand side of the inequality is only one. The constraint still works, however, it is not as strong as the one we used in 2ECSSP. Therefore, we try to develop two additional models to obtain a stronger formulation.

In the following formulation we use two different variables for the assignments of users to hubs and instead of y we use z and w. The aim of using two sets of variables is to improve the cut constraints by increasing the coefficients of the variables on the right hand side of constraint (5.4) from 1 to 2. We use x and t variables together with the z and w variables defined below to develop the following integer model.

$$z_{ij}(w_{ij}) = \begin{cases} 1, & \text{if } j \text{ is the first (second) node to which } i \text{ is assigned to,} \\ 0, & \text{otherwise} \end{cases}$$

$$\min \sum_{e\in E} c_e x_e + \sum_{(i,j)\in A} d_{ij}(w_{ij} + z_{ij}) + \sum_{i\in V\setminus\{0\}} d_{ii} t_i \qquad (5.5)$$

s.t.

$$\begin{aligned}
& t_i + \textstyle\sum_{j\in V\setminus\{i\}} z_{ij} = 1 \\
& t_i + \textstyle\sum_{j\in V\setminus\{i\}} w_{ij} = 1 && \forall i \in V \setminus \{0\} && (5.6)\\
& x_{ij} + w_{ij} + z_{ij} \le t_j && \forall (i,j) \in A && (5.7)\\
& x(\delta(S)) \ge 2\textstyle\sum_{j\in S\setminus\{i\}} z_{ij} + 2t_i \\
& x(\delta(S)) \ge 2\textstyle\sum_{j\in S\setminus\{i\}} w_{ij} + 2t_i && \forall S \subseteq V \setminus \{0\}, \forall i \in S && (5.8)\\
& x_e \in \{0,1\} && \forall e \in E \\
& z_{ij}, w_{ij} \in \{0,1\} && \forall (i,j) \in A :\ i \ne j \\
& t_i \in \{0,1\} && \forall i \in V \setminus \{0\}
\end{aligned}$$

This second model is called M2. Note that M1 and M2 are quite similar. The objective function (5.5) consists of the setup costs of the backbone and local access networks. Two set of constraints are required for the assignments and constraints (5.6) are used to imply that a node is either chosen as a hub or is assigned to another hub. Note that for two nodes $i, j \in V$, we have $z_{ij} + w_{ij} \le 1$ as a node cannot be assigned to the same node twice. This allows us to use the constraint (5.7) to impose that if a node i is assigned to a node j or the edge between i and j is used on the backbone network then j must be a hub. Constraint (5.8) is the same cut constraint with the one used for 2ECSSP. Since at most one assignment variable can take value 1 on the right hand side the coefficients of the assignment variables are equal to 2 as desired.

Unfortunately, M2 is not stronger than M1. Let P_1 and P_2 denote the feasible solution sets of the linear programming relaxations of M1 and M2, respectively. Let $(x, y, t) \in P_1$. Consider the solution (x, z, w, t) defined by $w_{ij} = z_{ij} = y_{ij}/2$ for every $(i, j) \in A$. It can be seen that $(x, z, w, t) \in P_2$ and both solutions have the same objective function value. Now let $(x, z, w, t) \in P_2$ and consider the solution (x, y, t) defined by $y_{ij} = w_{ij} + z_{ij}$ for every $(i, j) \in A$. We can observe that (x, y, t) is a solution in P_1 and the objective function values of the both solutions are equal. It is shown that a feasible solution in P_2 can be found with the same objective function value for every feasible solution of P_1 and vice versa. This shows that M_1 and M_2 are equivalent formulations.

Both M1 and M2 have exponential number of constraints. However, in M2 the number of variables is higher. The number of variables used in M2 can be reduced in the implementation, however, M2 will still have more variables. In addition, the number of separation problems that must be solved is also higher. Although the complexities are the same for both formulations, the number of solved problems would affect the solution performance.

Therefore we propose a third formulation, called M3, in which we keep track of the assignments in only one variable with three indices. In addition to x and t we define the following variable u.

$$u_{i\{j,k\}} = \begin{cases} 1, & \text{if } i \text{ is assigned to nodes } j \text{ and } k \\ 0, & \text{otherwise} \end{cases}$$

Let $E_i = \{\{j, k\} \in E \setminus \delta(i)\}$. We will use this set E to define the domains of some constraints.

$$\min \sum_{e\in E} c_e x_e + \sum_{i\in V\setminus\{0\}} d_{ii} t_i + \sum_{i\in V} \sum_{\{j,k\}\in E_i} (d_{ij} + d_{ik}) u_{i\{j,k\}} \tag{5.9}$$

$$\text{s.t. } t_i + \sum_{\{j,k\}\in E_i} u_{i\{j,k\}} = 1 \qquad \forall i \in V\setminus\{0\} \tag{5.10}$$

$$x_e + \sum_{k\in V:\{j,k\}\in E_i} u_{i\{j,k\}} \le t_j \qquad \forall (i,j)\in A, e=\{i,j\} \tag{5.11}$$

$$x_e + \sum_{k\in V:\{0,k\}\in E_i} u_{i\{0,k\}} \le 1 \qquad \forall i \in V\setminus\{0\}, e=\{i,0\} \tag{5.12}$$

$$x_e \le t_i \qquad \forall i \in V\setminus\{0\}, e=\{i,0\} \tag{5.13}$$

$$x(\delta(S)) \ge 2t_i + 2 \sum_{\{j,k\}\in E_i:|\{j,k\}\cap S|\ge 1} u_{i\{j,k\}} \qquad \forall S\subseteq V\setminus\{0\}, i\in S \tag{5.14}$$

$$x_e \in \{0,1\} \qquad \forall e\in E \tag{5.15}$$

$$u_{i\{j,k\}} \in \{0,1\} \qquad \forall i\in V\setminus\{0\}, \{j,k\}\in E_i \tag{5.16}$$

$$t_i \in \{0,1\} \qquad \forall i \in V\setminus\{0\}. \tag{5.17}$$

Here, the objective function (5.9) is equal to the cost of installing the backbone links, the hubs, and the access links. The assignment constraints (5.10) ensure that each node is a hub or it is assigned to two distinct nodes and the constraints (5.11), (5.12), and (5.13) ensure that nodes are assigned to hub nodes and backbone links can be installed between hub nodes. The cut constraints (5.14) model the requirement that the backbone network is 2-edge connected. Let $S \subseteq V \setminus \{0\}$ and $i \in S$. If i is a hub, i.e., $t_i = 1$, or if i is assigned to at least one hub in set S, i.e., $\sum_{\{j,k\}\in E_i:|\{j,k\}\cap S|\ge 1} u_{i\{j,k\}} = 1$, then the constraint becomes $x(\delta(S)) \ge 2$. Otherwise, i is assigned to two hubs in $V \setminus S$ and the constraint is redundant. Constraints (5.16)-(5.17) state that the variables are binary.

We first remark that for $S \subseteq V \setminus \{0\}$ and $i \in S$, when i is assigned to one hub in S and one hub in $V \setminus S$, the right hand side of the cut constraint (5.14) of the three-index model is equal to 2 (hence this constraint imposes the installation of minimum two edges on the cut between S and $V \setminus S$), whereas the cut constraint (5.4) of the two-index formulation has right hand side equal to 1. Now we can compare the strength of the LP relaxations of M1 and M3.

Theorem 5.1 *M3 is stronger than M1.*

Proof First, note that $y_{ij} = \sum_{k \in V:\{j,k\} \in E_i} u_{i\{j,k\}} = \sum_{k \in V \setminus \{i,j\}} u_{i\{j,k\}}$ for all $(i,j) \in A$. Using this equivalence, we can show that the objective functions, the assignment and relation constraints are the same in both formulations. Now, we study the cut constraints. Let $S \subseteq V \setminus \{0\}$ and $i \in S$. The right hand side of constraint (5.14) is equal to

$$\begin{aligned} 2t_i + 2 \sum_{\{j,k\} \in E_i : |\{j,k\} \cap S| \geq 1} u_{i\{j,k\}} &= 2t_i + \sum_{j \in S \setminus \{i\}} \sum_{k \in V \setminus \{i,j\}} u_{i\{j,k\}} + \sum_{\{j,k\} \in E_i \cap \delta(S)} u_{i\{j,k\}} \\ &= 2t_i + \sum_{j \in S \setminus \{i\}} y_{ij} + \sum_{\{j,k\} \in E_i \cap \delta(S)} u_{i\{j,k\}}, \end{aligned}$$

which is greater than or equal to the right hand side of the cut constraint (5.4). Thus the three-index formulation M3 is stronger than the two-index formulation M1. □

Even though M3 is stronger, our preliminary tests have shown that it is not advantageous to use it in a branch and cut algorithm as it takes much longer time to solve its LP-relaxations. In addition, we could not find a polynomial algorithm to solve the separation problem associated with the constraints (5.14). Although this constraint is based on the cuts, it is not possible to solve it easily using minimum cut algorithms.

Since the LP's of M3 is large and hard to solve, and the separation problem of the cut constraints is difficult, we use M1 to find the optimal solution of 2ECSDHP. However, it is possible to make M1 stronger by using some valid inequalities obtained from the projection of M3. In the following subsection we provide the details of the projection.

5.2.1 Projection inequalities

Given (x, t, y), there exists a vector u that satisfies

$$\sum_{\{j,k\} \in E_i \cap \delta(S)} u_{i\{j,k\}} \leq x(\delta(S)) - 2t_i - \sum_{j \in S \setminus \{i\}} y_{ij} \quad \forall S \subseteq V \setminus \{0\}, i \in S \tag{5.18}$$

$$\sum_{k \in V : \{j,k\} \in E_i} u_{i\{j,k\}} = y_{ij} \quad \forall (i,j) \in A \tag{5.19}$$

$$u_{i\{j,k\}} \geq 0 \quad \forall i \in V \setminus \{0\}, \{j,k\} \in E_i \tag{5.20}$$

if and only if

$$\sum_{i \in V \setminus \{0\}} \sum_{S \subseteq V \setminus \{0\} : i \in S} \left(x(\delta(S)) - 2t_i - \sum_{j \in S \setminus \{i\}} y_{ij} \right) \alpha_{iS} + \sum_{(i,j) \in A} \beta_{ij} y_{ij} \geq 0 \tag{5.21}$$

for all vectors (α, β) such that

$$\sum_{S \subseteq V \setminus \{0\} : |S \cap \{j,k\}| = 1} \alpha_{iS} + \beta_{ij} + \beta_{ik} \geq 0 \quad \forall i \in V \setminus \{0\}, \{j,k\} \in E_i \tag{5.22}$$

$$\alpha_{iS} \geq 0 \quad \forall S \subseteq V \setminus \{0\}, i \in S. \tag{5.23}$$

First notice that the above system can be disaggregated for each node $i \in V \setminus \{0\}$. For a given $i \in V \setminus \{0\}$, consider (α, β) where $\beta = 0$, $\alpha_{iS} = 1$ for some $S \subseteq V \setminus \{0\}$ with $i \in S$ and all other entries are zero. The projection inequality (5.21) for this choice of (α, β) is the same as the cut inequality (5.4). This gives an alternative proof to Theorem 5.1.

Now consider again a vector (α, β) where $\alpha_{iS} = 1$ for some $S \subseteq V \setminus \{0\}$ with $i \in S$ and all other entries of α are zero. One feasible β vector is as follows: $\beta_{ij} = -1$ for some $j \in S \setminus \{i\}$, $\beta_{ik} = 1$ for all $k \in S \setminus \{i, j\}$, and other entries of β are zero. This yields the projection inequality

$$x(\delta(S)) \geq 2t_i + 2y_{ij}. \tag{5.24}$$

Next we again consider a vector (α, β) where $\alpha_{iS} = 1$ for some $S \subseteq V \setminus \{0\}$ with $i \in S$ and all other entries of α are zero. Let $j \in V \setminus S$. Consider the vector

β where $\beta_{ij} = -1$, $\beta_{ik} = 1$ for all $k \in V \setminus (S \cup \{j\})$, and other entries of β are zero. The resulting projection inequality (5.21) is

$$x(\delta(S)) \geq 2t_i + \sum_{k \in S \setminus \{i\}} y_{ik} + y_{ij} - \sum_{k \in V \setminus (S \cup \{j\})} y_{ik}. \tag{5.25}$$

We conclude this section with the following remark. If we replace the cut constraints (5.4) with projection inequalities (5.24) or with (5.25) in the two-index formulation, we obtain two alternative formulations for DH. It is not possible to compare these two-index formulations among themselves (later, we prove that all three families of inequalities (5.4), (5.24), and (5.25) are facet defining under some conditions). However, we can conclude that the three-index formulation is stronger than these two new formulations as inequalities (5.24) and (5.25) are projection inequalities.

5.3 Polyhedral Analysis

In this section, we provide the polyhedral analysis done for the 2ECSDHP. We also present some families of valid inequalities.

We start by modifying M1 to obtain a full dimensional polytope as we did for the 2ECSSP. We eliminate the variables t_i for $i \in V \setminus \{0\}$. From constraints (5.2) we make the substitution $t_i = 1 - \frac{\sum_{j \in V \setminus \{i\}} y_{ij}}{2}$ for $i \in V \setminus \{0\}$. This yields:

$$z = \sum_{i \in V\setminus\{0\}} d_{ii} + \min \sum_{e \in E} c_e x_e + \sum_{(i,j)\in A} d'_{ij} y_{ij}$$

$$\text{s.t. } 2x_e + 2y_{ij} + \sum_{k \in V\setminus\{j\}} y_{jk} \le 2 \quad \forall (i,j) \in A : j \neq 0, e = \{i,j\} \tag{5.26}$$

$$x_e + y_{i0} \le 1 \quad \forall i \in V \setminus \{0\}, e = \{i,0\} \tag{5.27}$$

$$2x_e + \sum_{k \in V\setminus\{i\}} y_{ik} \le 2 \quad \forall i \in V \setminus \{0\}, e = \{i,0\} \tag{5.28}$$

$$x(\delta(S)) + \sum_{j \in V\setminus S} y_{ij} \ge 2 \quad \forall S \subseteq V \setminus \{0\}, i \in S \tag{5.29}$$

$$x_e \in \{0,1\} \quad \forall e \in E \tag{5.30}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in A \tag{5.31}$$

where $d'_{ij} = d_{ij} - \frac{d_{ii}}{2}$ for $(i,j) \in A$.

To show that this formulation is equivalent to M1 of Section 5.2, we need to ensure that $\sum_{j\in V\setminus\{i\}} y_{ij} \in \{0,2\}$ and so $t_i \in \{0,1\}$ for all $i \in V \setminus \{0\}$.

Let $i \in V \setminus \{0\}$. If $x(\delta(i)) > 0$, then constraints (5.26), (5.27), (5.28), and (5.31) imply that $\sum_{j\in V\setminus\{i\}} y_{ij} = 0$. On the other hand, if $x(\delta(i)) = 0$, the cut constraint (5.29) for $S = \{i\}$ implies that $\sum_{j\in V\setminus\{i\}} y_{ij} \ge 2$. As $\sum_{j\in V\setminus\{i\}} y_{ij} \le 2$ due to constraints (5.28), we have $\sum_{j\in V\setminus\{i\}} y_{ij} = 2$. Therefore, $\sum_{j\in V\setminus\{i\}} y_{ij} \in \{0,2\}$ in any feasible solution to the above model.

Let $X = \{(x,y) \in R^{|E|+|A|} : (x,y)$ satisfies (5.26)-(5.31)$\}$ and $\mathcal{P} = conv(X)$.

We define the following vectors. For an edge $e \in E$, let χ_e be a vector of size $|E|$, where the entry corresponding to edge e is equal to 1 and the remaining entries are 0. Similarly, for $i \in V \setminus \{0\}$ and distinct nodes $j,k \in V \setminus \{i\}$, we define γ_{ijk} to be a vector of size $|A|$, where the entries corresponding to arcs (i,j) and (i,k) are equal to 1 and the other entries are equal to 0. Note that these vectors are similar to the ones defined in Chapter 3, however, γ vector has two positive components and it is not a unit vector unlike the γ vector defined during the polyhedral analysis of 2ECSSP polytope. We assume that $|V| \geq 7$ in the rest of the chapter.

5.3.1 Dimension and trivial facets

First, we show that the polytope $\mathcal{P}$ is full dimensional.

Theorem 5.2 $\mathcal{P}$ *is full dimensional.*

Proof Suppose that all solutions $(x, y) \in \mathcal{P}$ satisfy $ax + by = \beta$. Let $e' \in E$ and consider the solutions $(\sum_{e \in E} \chi_e, 0)$ and $(\sum_{e \in E \setminus \{e'\}} \chi_e, 0)$. Both solutions are in $\mathcal{P}$. So we have $a_e^{'} = 0$.

Let $(i, j) \in A$ with $j \neq 0$. Consider the solutions $(\sum_{e \in E} \chi_e, 0)$ and $(\sum_{e \in E \setminus \delta(i)} \chi_e, \gamma_{ij0})$. As both solutions are in $\mathcal{P}$ and $a = 0$, we have $b_{ij} + b_{i0} = 0$.

Let $i \in V \setminus \{0\}$ and $j, k \in V \setminus \{i, 0\}$ be distinct nodes and consider solutions $(\sum_{e \in E \setminus \delta(i)} \chi_e, \gamma_{ij0})$ and $(\sum_{e \in E \setminus \delta(i)} \chi_e, \gamma_{ijk})$. These solutions are in $\mathcal{P}$ implying that $b_{ik} = b_{i0}$. As we also have $b_{ij} + b_{i0} = 0$, we can conclude that $b_{ij} = b_{i0} = 0$.

Now as both a and b are zero, β is also zero. So no inequality is satisfied at equality by all points of $\mathcal{P}$. $\square$

The next two theorems are on the strength of the nonnegativity constraints.

Theorem 5.3 *For* $e \in E$*, inequality* $x_e \geq 0$ *is facet defining for* $\mathcal{P}$*.*

Proof Let $e \in E$ and $\mathcal{F} = \{(x, y) \in \mathcal{P} : x_e = 0\}$. Suppose that all solutions $(x, y) \in \mathcal{F}$ satisfy $ax + by = \beta$. Let $e^{'} \in E \setminus \{e\}$ and consider the solutions $(\sum_{\hat{e} \in E \setminus \{e\}} \chi_{\hat{e}}, 0)$ and $(\sum_{\hat{e} \in E \setminus \{e, e'\}} \chi_{\hat{e}}, 0)$. Both solutions are in $\mathcal{F}$, so $a_{e^{'}} = 0$.

Let $i \in V \setminus \{0\}$ and $j, k, l \in V \setminus \{i\}$ be distinct nodes. As the solutions $(\sum_{\hat{e} \in E \setminus (\delta(i) \cup \{e\})} \chi_{\hat{e}}, \gamma_{ijk})$ and $(\sum_{\hat{e} \in E \setminus (\delta(i) \cup \{e\})} \chi_{\hat{e}}, \gamma_{ijl})$ are both in $\mathcal{F}$, we have $b_{ik} = b_{il}$. So $b_{ij} = \theta_i$ for some $\theta_i \in \mathbb{R}$ for all $(i,j) \in A$. Now consider the solutions $(\sum_{\hat{e} \in E \setminus \{e\}} \chi_{\hat{e}}, 0)$ and $(\sum_{\hat{e} \in E \setminus (\delta(i) \cup \{e\})} \chi_{\hat{e}}, \gamma_{ijk})$. These solutions are both in $\mathcal{F}$ and $a_{\hat{e}} = 0$ for all $\hat{e} \in E \setminus \{e\}$, and hence, $2\theta_i = 0$. Finally, since the solution $(0,0)$ is in $\mathcal{F}$, $\beta = 0$.

Since all the coefficients except a_e are zero, $ax + by = \beta$ is a positive multiple of $x_e = 0$, and hence $x_e \geq 0$ defines a facet of $\mathcal{P}$. $\square$

Theorem 5.4 *For $(i,j) \in A$, inequality $y_{ij} \geq 0$ is facet defining for $\mathcal{P}$.*

Proof Let $(i,j) \in A$ and $\mathcal{F} = \{(x,y) \in \mathcal{P} : y_{ij} = 0\}$. Suppose that all solutions $(x,y) \in \mathcal{F}$ satisfy $ax + by = \beta$. Let $e' \in E$ and consider the solutions $(\sum_{e \in E} \chi_e, 0)$ and $(\sum_{e \in E \setminus \{e'\}} \chi_e, 0)$. Since both solutions are in $\mathcal{F}$, we have $a_{e'} = 0$.

Let $u \in V \setminus \{0\}$. If $u \neq i$, then let $v, k, l \in V \setminus \{u\}$ be distinct nodes. If $u = i$, then let $v, k, l \in V \setminus \{i, j\}$ be distinct nodes. The solutions $(\sum_{e \in E \setminus \delta(u)} \chi_e, \gamma_{uvk})$ and $(\sum_{e \in E \setminus \delta(u)} \chi_e, \gamma_{uvl})$ are in $\mathcal{F}$. So we can conclude $b_{uv} = \theta_u$ for some $\theta_u \in \mathbb{R}$ for all $(u,v) \in A \setminus \{(i,j)\}$. Considering the solutions $(\sum_{e \in E} \chi_e, 0)$ and $(\sum_{e \in E \setminus \delta(u)} \chi_e, \gamma_{uvk})$, it can be seen that $2\theta_u = 0$. Moreover, $\beta = 0$ since $(0,0)$ is in $\mathcal{F}$.

Since all the coefficients except b_{ij} are zero, $ax + by = \beta$ is a positive multiple of $y_{ij} = 0$ and $y_{ij} \geq 0$ is facet defining for $\mathcal{P}$. $\square$

The inequalities $x_e \leq 1$ for $e \in E$ and $y_{ij} \leq 1$ for $(i,j) \in A$ are not facet defining as they are implied by constraints (5.26), (5.27), and (5.28).

Next, we study the constraints (5.26) and (5.28).

Theorem 5.5 *Let $(i,j) \in A$ such that $j \neq 0$ and $e = \{i,j\}$. The inequality (5.26) defines a facet of $\mathcal{P}$.*

Proof Let $(i, j) \in A$ such that $j \neq 0$, $e = \{i, j\}$, and $\mathcal{F} = \{(x, y) \in \mathcal{P} : 2x_e + 2y_{ij} + \sum_{k \in V \setminus \{j\}} y_{jk} = 2\}$. Suppose that every solution (x, y) in $\mathcal{F}$ also satisfies $ax + by = \beta$.

Let $e^{'} \in E \setminus \{e\}$. As $(\sum_{\hat{e} \in E} \chi_{\hat{e}}, 0)$ and the solution $(\sum_{\hat{e} \in E \setminus \{e^{'}\}} \chi_{\hat{e}}, 0)$ are both in $\mathcal{F}$, we have $a_{e^{'}} = 0$.

Let $k \in V \setminus \{0, i, j\}$ and $u, v, w \in V \setminus \{k\}$ be distinct nodes. The solutions $(\sum_{\hat{e} \in E \setminus \delta(k)} \chi_{\hat{e}}, \gamma_{kuv})$ and $(\sum_{\hat{e} \in E \setminus \delta(k)} \chi_{\hat{e}}, \gamma_{kuw})$ are in $\mathcal{F}$. This implies that $b_{kl} = \theta_k$ for some $\theta_k \in \mathbb{R}$. As both $(\sum_{\hat{e} \in E \setminus \delta(k)} \chi_{\hat{e}}, \gamma_{kuv})$ and $(\sum_{\hat{e} \in E} \chi_{\hat{e}}, 0)$ are in $\mathcal{F}$ and $a_{\hat{e}} = 0$ for all $\hat{e} \in E \setminus \{e\}$, we have $2\theta_k = 0$. Therefore $b_{kl} = 0$ for every $(k, l) \in A$ such that $k \in V \setminus \{0, i, j\}$.

Let $u, v, w \in V \setminus \{i, j\}$ be distinct nodes. We define $\overline{x} = \sum_{\hat{e} \in E \setminus (\delta(i) \cup \delta(j))} \chi_{\hat{e}}$. Consider the solutions $(\overline{x}, \gamma_{iuv} + \gamma_{juv})$ and $(\overline{x}, \gamma_{iuw} + \gamma_{juv})$. As these solutions are in $\mathcal{F}$ we conclude that $b_{il} = \theta_i$ for some $\theta_i \in \mathbb{R}$ for every $l \in V \setminus \{i, j\}$. In addition we can see that, $(\overline{x}, \gamma_{iuv} + \gamma_{juv})$ and $(\sum_{\hat{e} \in E \setminus \delta(j)} \chi_{\hat{e}}, \gamma_{juv})$ are in $\mathcal{F}$. As $a_{\hat{e}} = 0$ for all $\hat{e} \in E \setminus \{e\}$ we have $2\theta_i = 0$. Therefore $b_{il} = 0$ for every $(i, l) \in A$ for all $l \in V \setminus \{i, j\}$.

Next consider the solutions $(\sum_{\hat{e} \in E \setminus \delta(j)} \chi_{\hat{e}}, \gamma_{juv})$ and $(\sum_{\hat{e} \in E \setminus \delta(j)} \chi_{\hat{e}}, \gamma_{juw})$ where $u, v, w \in V \setminus \{j\}$ are distinct nodes. Both solutions are in $\mathcal{F}$ implying that $b_{jl} = \sigma$ for some $\sigma \in \mathbb{R}$ for every $l \in V \setminus \{j\}$. Note that $(\sum_{\hat{e} \in E \setminus \delta(i)} \chi_{\hat{e}}, \gamma_{ij0})$ is also in $\mathcal{F}$. So we can say that $b_{ij} = 2\sigma$ since $a_{\hat{e}} = 0$ for all $\hat{e} \in E \setminus \{e\}$, and $b_{i0} = 0$. Finally, consider the solution $(\sum_{\hat{e} \in E} \chi_{\hat{e}}, 0) \in \mathcal{F}$. We know that $a_{\hat{e}} = 0$ for all $\hat{e} \in E \setminus \{e\}$ and $b_{i0} = 0$. Therefore $a_e = 2\sigma$. Also $\beta = 2\sigma$.

Therefore, $ax + by = \beta$ is a σ multiple of $2x_e + 2y_{ij} + \sum_{k \in V \setminus \{j\}} y_{jk} = 2$ and $\mathcal{F}$ is a facet of $\mathcal{P}$. $\square$

Let $i \in V \setminus \{0\}$ and $e = \{i, 0\}$. Inequality (5.27) is not facet defining since all feasible solutions that satisfy $x_e + y_{i0} = 1$ also satisfy $y_{i0} = \sum_{k \in V \setminus \{0, i\}} y_{ik}$.

Theorem 5.6 *Let $i \in V \setminus \{0\}$ and $e = \{i, 0\}$. The inequality (5.28) defines a facet of $\mathcal{P}$.*

Proof Similar to the proof of Theorem 5.5. □

The remaining constraints of our formulation are the cut constraints (5.29). We investigate the conditions under which these inequalities define facets in Section 5.3.3.

5.3.2 Valid inequalities involving only the assignment variables

In this section, we present two families of valid inequalities that involve only the assignment variables.

The first valid inequality we present in this part is the triangle inequality which is also shown to be valid and facet defining for the 2ECSSP polytope.

Let $i, j, k \in V \setminus \{0\}$ be distinct nodes. Then the following triangle inequality is valid for $\mathcal{P}$.

$$y_{ij} + y_{jk} + y_{ki} \leq 1 \qquad\qquad i, j, k \in V \setminus \{0\} \qquad (5.32)$$

Theorem 5.7 *Let $i, j, k \in V \setminus \{0\}$ be distinct nodes. Inequality (5.32) defines a facet of $\mathcal{P}$.*

Proof Let $\mathcal{F} = \{(x, y) \in \mathcal{P} : y_{ij} + y_{jk} + y_{ki} = 1\}$. Suppose that every solution (x, y) in $\mathcal{F}$ also satisfies $ax + by = \beta$.

Let $e' \in E \setminus \delta(i)$ and consider the solutions $(\sum_{e \in E \setminus \delta(i)} \chi_e, \gamma_{ij0})$ and $(\sum_{e \in E \setminus \delta(u)} \chi_e - \chi_{e'}, \gamma_{ij0})$. They are both in $\mathcal{F}$ implying that $a_e = 0$ for all $e \in E \setminus \delta(i)$. Now let $e' \in \delta(i) \setminus \{\{i, j\}\}$ and consider the solutions $(\sum_{e \in E \setminus \delta(j)} \chi_e, \gamma_{jk0})$ and $(\sum_{e \in E \setminus \delta(j)} \chi_e - \chi_{e'}, \gamma_{jk0})$. Since they are both in $\mathcal{F}$, we have $a_e = 0$ for all $e \in \delta(i) \setminus \{\{i, j\}\}$. Finally, considering the solutions $(\sum_{e \in E \setminus \delta(k)} \chi_e, \gamma_{ki0})$ and $(\sum_{e \in E \setminus \delta(k)} \chi_e - \chi_{ij}, \gamma_{ki0})$ in $\mathcal{F}$, we obtain $a_e = 0$ for all $e \in E$.

Let $l \in V \setminus \{0, i, j, k\}$ and $u, v, w \in V \setminus \{l, i\}$. Clearly, $(\sum_{e \in E \setminus (\delta(i) \cup \delta(l))} \chi_e, \gamma_{ij0} + \gamma_{luv})$ is a solution in $\mathcal{F}$. Observe that the solution $(\sum_{e \in E \setminus (\delta(i) \cup \delta(l))} \chi_e, \gamma_{ij0} + \gamma_{luw})$ is also in $\mathcal{F}$. This shows that $b_{lu} = \theta_l$ for some $\theta_l \in \mathbb{R}$ for $(l, u) \in A$. Moreover, $(\sum_{e \in E \setminus \delta(i)} \chi_e, \gamma_{ij0})$ is also a solution in $\mathcal{F}$. So we have $b_{lu} = 0$ for all $(l, u) \in A$ such that $l \in V \setminus \{0, i, j, k\}$ and $u \in V \setminus \{l, i\}$. In a similar way, we can construct solutions $(\sum_{e \in E \setminus (\delta(j) \cup \delta(l))} \chi_e, \gamma_{jk0} + \gamma_{li0})$ and $(\sum_{e \in E \setminus \delta(j)} \chi_e, \gamma_{jk0})$. As both solutions are in $\mathcal{F}$, $a_e = 0$ for $e \in E$, and $b_{l0} = 0$, we conclude that $b_{li} = 0$. So we have $b_{lu} = 0$ for every $(l, u) \in A$ such that $l \in V \setminus \{0, i, j, k\}$.

Let $u, v, w \in V \setminus \{i, j\}$. Clearly, $(\sum_{e \in E \setminus (\delta(i) \cup \delta(j))} \chi_e, \gamma_{iuv} + \gamma_{jk0})$ is a solution in $\mathcal{F}$. Observe that the solution $(\sum_{e \in E \setminus (\delta(i) \cup \delta(j))} \chi_e, \gamma_{jk0} + \gamma_{iuw})$ is also in $\mathcal{F}$. Moreover, $(\sum_{e \in E \setminus \delta(j)} \chi_e, \gamma_{jk0})$ is also a solution in $\mathcal{F}$. By symmetry, similar solutions can be constructed for nodes j and k. Therefore, we have $b_{uv} = 0$ for all $(u, v) \in A \setminus \{(i, j), (j, k), (k, i)\}$.

Finally consider the solutions $(\sum_{e \in E \setminus \delta(i)} \chi_e, \gamma_{ij0})$, $(\sum_{e \in E \setminus \delta(j)} \chi_e, \gamma_{jk0})$, and $(\sum_{e \in E \setminus \delta(k)} \chi_e, \gamma_{ki0})$. As they are all in $\mathcal{F}$, we have $b_a = \sigma$ for some $\sigma \in \mathbb{R}$ for $a \in \{(i, j), (j, k), (k, i)\}$.

So $ax + by = \beta$ is a multiple of $y_{ij} + y_{jk} + y_{ki} = 1$ and $\mathcal{F}$ is a facet of $\mathcal{P}$. □

The next family of valid inequalities uses the idea of dual homing. If a node $i \in V \setminus \{0\}$ is assigned to a hub, say $j \in V \setminus \{i\}$, then it is not a hub node and must be assigned to a second hub node. This yields the following valid inequality:

$$y_{ij} \leq \sum_{k \in V \setminus \{i,j\}} y_{ik}. \tag{5.33}$$

Theorem 5.8 *For $(i, j) \in A$, inequality (5.33) defines a facet of $\mathcal{P}$.*

Proof Let $(i, j) \in A$ and $\mathcal{F} = \{(x, y) \in \mathcal{P} : y_{ij} - \sum_{k \in V \setminus \{i,j\}} y_{ik} = 0\}$. Suppose that every solution (x, y) in $\mathcal{F}$ also satisfies $ax + by = \beta$.

Let $e \in E$. Consider the solutions $(\sum_{\hat{e} \in E} \chi_{\hat{e}}, 0)$ and $(\sum_{\hat{e} \in E \setminus \{e\}} \chi_{\hat{e}}, 0)$. As they are both in $\mathcal{F}$, we have $a_e = 0$.

Let $k \in V \backslash \{0, i\}$ and $u, v, w \in V \backslash \{k\}$ be distinct nodes. Considering the solutions $(\sum_{e \in E \backslash \delta(k)} \chi_e, \gamma_{kuv})$ and $(\sum_{e \in E \backslash \delta(k)} \chi_e, \gamma_{kuw})$, which are both in $\mathcal{F}$, we obtain $b_{kl} = \theta_k$ for some $\theta_k \in \mathbb{R}$ for every $(k, l) \in A$. In addition, $(\sum_{e \in E \backslash \delta(k)} \chi_e, \gamma_{kuv})$ and $(\sum_{e \in E} \chi_e, 0)$ are both in $\mathcal{F}$. Since $a_e = 0$ for all $e \in E$, we have $2\theta_k = 0$, and hence $b_{kl} = 0$ for every $(k, l) \in A$ such that $k \in V \setminus \{0, i\}$.

Let $u, v \in V \setminus \{i, j\}$ be distinct nodes. The solutions $(\sum_{e \in E \backslash \delta(i)} \chi_e, \gamma_{iju})$ and $(\sum_{e \in E \backslash \delta(i)} \chi_e, \gamma_{ijv})$ are both in $\mathcal{F}$. This shows that $b_{il} = \sigma$ for some $\sigma \in \mathbb{R}$ for every $l \in V \setminus \{i, j\}$. Finally, we consider the solutions $(\sum_{e \in E \backslash \delta(i)} \chi_e, \gamma_{iju})$ and $(\sum_{e \in E} \chi_e, 0)$. Both solutions are in $\mathcal{F}$ and note that we know $a_e = 0$ for all $e \in E$ and $b_{iu} = \sigma$. Together these imply that $b_{ij} = -\sigma$. It is easy to see that $\beta = 0$.

Now we can conclude that $ax + by = \beta$ is a multiple of $y_{ij} - \sum_{k \in V \backslash \{i,j\}} y_{ik} = 0$ and $\mathcal{F}$ is a facet of $\mathcal{P}$. $\square$

5.3.3 Valid inequalities involving cuts

In this section, we investigate the strength of several families of valid inequalities based on cuts. These inequalities impose lower bounds on the number of edges to be installed between two nodes sets S and $V \setminus S$. Here we need to point out a difference between the 2ECSSP and 2ECSDHP. Note that, the number of nodes in a graph is either equal to 1 or greater than 2. However, in 2ECSDHP there must be at least three hubs in the backbone network while the backbone of the 2ECSSP can be composed of a single hub. This is because of the dual homing architecture. If the backbone has only one hub, then there are some users and this makes it necessary to have at least two hubs in the backbone which is a contradiction. Therefore if $|V \setminus S| \leq 2$ then we know that there is at least one hub in S. This yields the following inequality,

$$x(\delta(S)) \geq 2 \qquad \forall S \subset V \setminus \{0\} : |V \setminus S| \leq 2. \tag{5.34}$$

which is clearly stronger than the cut inequalities (5.4) when $|V \setminus S| \leq 2$.

The next theorem gives necessary and sufficient conditions for the cut constraints (5.29) to be facet defining for $\mathcal{P}$.

Theorem 5.9 *Let $S \subseteq V \setminus \{0\}$ such that $S \neq \emptyset$ and $i \in S$. Inequality (5.29) defines a facet of $\mathcal{P}$ if and only if the following conditions are satisfied together:*

i. $|V \setminus S| \geq 3$

ii. $|S| \geq 4$ *or* $|S| = 1$.

Proof Let $\mathcal{F} = \{(x, y) \in \mathcal{P} : x(\delta(S)) + \sum_{j \in V \setminus S} y_{ij} = 2\}$. Suppose that every solution (x, y) in $\mathcal{F}$ also satisfies $ax + by = \beta$.

Let $S \subseteq V \setminus \{0\}$ such that $S \neq \emptyset$ and $i \in S$. Suppose that $|V \setminus S| \leq 2$. Since a terminal node is assigned to two hubs and the backbone network is to be 2-edge connected, there must be at least three hubs on the backbone network. This is because the graph does not contain multiple edges. So for the cases $|V \setminus S| \leq 3$ there is at least one hub in set S. Then inequality (5.29) is dominated by the valid inequality $x(\delta(S)) \geq 2$. Similarly, if $|S| = 2$ inequality (5.36) dominates (5.29), which can be represented with $x(\delta(S)) \geq 2t_i + y_{ij}$ using the t variables. Let $i, k, l \in V \setminus \{0\}$ be distinct nodes and $S = \{i, k, l\}$. Let $(x, y) \in X$ with $x(\delta(S)) + \sum_{j \in V \setminus S} y_{ij} = 2$. If i is a hub node, then $y_{ik} = y_{il} = 0$. Now suppose that node i is not a hub. If $x(\delta(S)) = 2$, then $\sum_{j \in V \setminus S} y_{ij} = 0$ and i must be assigned to nodes k and l. So $y_{ik} = y_{il} = 1$. Finally, if $x(\delta(S)) = 0$, then $\sum_{j \in V \setminus S} y_{ij} = 2$, and hence $\sum_{j \in S} y_{ij} = 0$ implying $y_{il} = y_{ik} = 0$. Therefore, all solutions $(x, y) \in X$ with $x(\delta(S)) + \sum_{j \in V \setminus S} y_{ij} = 2$ also satisfy $y_{ik} = y_{il}$ and the cut inequality is not facet defining.

Now suppose that $|S| \geq 4$ and $|V \setminus S| \geq 3$. Notice that as G is complete, $G(S)$ and $G(V \setminus S)$ are 2-edge connected.

Let e_1, e_2 be any two edges in $\delta(S)$ and $x = \sum_{e \in E(S) \cup E(V \setminus S)} \chi_e$. Then $(x + \chi_{e_1} + \chi_{e_2}, 0)$ is a solution in $\mathcal{F}$. Let $e' \in \delta(S) \setminus \{e_1, e_2\}$. As $(x + \chi_{e_1} + \chi_{e_2}, 0)$ and the solutions $(x + \chi_{e_1} + \chi_{e'}, 0)$ and $(x + \chi_{e_2} + \chi_{e'}, 0)$ are all in $\mathcal{F}$, we have $a_{e_1} = a_{e_2} = a_{e'}$. Therefore $a_e = \sigma$ for all $e \in \delta(S)$ for some $\sigma \in \mathbb{R}$.

Let $e' \in E(S)$ and e_1 and e_2 be two edges in $\delta(S)$ incident to the two endpoints of e' such that $e_1 \cap e_2 \cap e' = \emptyset$. Consider the solutions $(x + \chi_{e_1} + \chi_{e_2}, 0)$ and $(x + \chi_{e_1} + \chi_{e_2} - \chi_{e'}, 0)$. As they are both in $\mathcal{F}$, we have $a_{e'} = 0$. We can show similarly that $a_{e'} = 0$ for all $e' \in E(V \setminus S)$.

Let $j \in V \setminus \{i, 0\}$ and e_1, e_2 be two edges in $\delta(S) \setminus \delta(j)$ with different endpoints in S if $j \in S$ and with different endpoints in $V \setminus S$ if $j \in V \setminus S$. We define $\overline{x} = x - \sum_{e \in \delta(j)} x_e \chi_e$. Let $u, v, w \in V \setminus \{j\}$. Clearly, the solution $(\overline{x} + \chi_{e_1} + \chi_{e_2}, \gamma_{juv})$ is in $\mathcal{F}$. Note that $(\overline{x} + \chi_{e_1} + \chi_{e_2}, \gamma_{juw})$ is also in $\mathcal{F}$. So we have $b_{jk} = \beta_j$ for some $\beta_j \in \mathbb{R}$ for every $k \in V \setminus \{j\}$. Moreover, $(x + \chi_{e_1} + \chi_{e_2}, 0)$ is also in $\mathcal{F}$. As $a_e = 0$ for all $e \in E(S) \cup E(V \setminus S)$, we have $2\beta_j = 0$. Therefore $b_{jk} = 0$ for every $(j, k) \in A$ such that $j \neq i$.

Let e_1, e_2 be two edges in $\delta(S) \setminus \delta(i)$ with different endpoints in S. Let $\hat{x} = x - \sum_{e \in \delta(i)} x_e \chi_e$. Let $u, v, w \in S \setminus \{i\}$. Consider the solution $(\hat{x} + \chi_{e_1} + \chi_{e_2}, \gamma_{iuv})$, which is in $\mathcal{F}$. Observe that, $(\hat{x} + \chi_{e_1} + \chi_{e_2}, \gamma_{iuw})$ is also in $\mathcal{F}$. So we have $b_{ik} = \theta$ for some $\theta \in \mathbb{R}$ for every $k \in S \setminus \{i\}$. Besides, we have $(x + \chi_{e_1} + \chi_{e_2}, 0)$ in $\mathcal{F}$. As $a_e = 0$ for all $e \in E(S) \cup E(V \setminus S)$, we have $2\theta = 0$. So we can conclude that $b_{ik} = 0$ for all $k \in S \setminus \{i\}$.

Let $e_1, e_2 \in \delta(S)$ and $u, v, w \in V \setminus S$. We define $x' = \sum_{e \in E(V \setminus S)} \chi_e$ and $y' = \sum_{j \in S} \gamma_{juv}$. Now we can see that the solutions (x', y') and $(x', y' - \gamma_{iuv} + \gamma_{iuw})$ are both in $\mathcal{F}$. So we have $b_{ik} = \xi$ for some $\xi \in \mathbb{R}$ for all $k \in V \setminus S$. Moreover, $(x + \chi_{e_1} + \chi_{e_2}, 0)$ is also in $\mathcal{F}$. As $a_e = 0$ for all $e \in E(S)$, $b_{jk} = 0$ for all $(j, k) \in A$ with $j \neq i$, and $a_e = \sigma$ for all $e \in \delta(S)$ we have $2\xi = 2\sigma$. Therefore $b_{ik} = \sigma$ for all $k \in V \setminus S$.

Finally, as the solution $(x + \chi_{e_1} + \chi_{e_2}, 0)$ is in $\mathcal{F}$, $\beta = 2\sigma$. Consequently, $ax + by = \beta$ is a σ multiple of $x(\delta(S)) + \sum_{j \in V \setminus S} y_{ij} = 2$ and $\mathcal{F}$ is a facet of $\mathcal{P}$.

The proof for the case with $|S| = 1$ can be done in a similar way. $\square$

Similar to the triangle inequality we can also show that the cut inequality based on the triangle inequality is also valid and facet defining for $\mathcal{P}$.

Let $S \subseteq V \setminus \{0\}$ and i, j, k be distinct nodes of S and consider the following inequality.

$$x(\delta(S)) \geq 2(y_{ij} + y_{jk} + y_{ki}) \qquad S \subseteq V \setminus \{0\},\ i, j, k \in S \tag{5.35}$$

The next theorem gives the conditions under which the inequality (5.35) defines a facet of $\mathcal{P}$.

Theorem 5.10 *Let $S \subseteq V \setminus \{0\}$ and $i, j, k \in S$ be distinct nodes. If $|S| \geq 4$ and $|V \setminus S| = 1$ or $|V \setminus S| \geq 3$ then inequality (5.35) defines a facet of $\mathcal{P}$.*

Proof Let $\mathcal{F} = \{(x, y) \in \mathcal{P} : x(\delta(S)) = 2(y_{ij} + y_{jk} + y_{ki})\}$. Suppose that every solution (x, y) in $\mathcal{F}$ also satisfies $ax + by = \beta$. We will show that $ax + by = \beta$ is a multiple of $x(\delta(S)) = 2(y_{ij} + y_{jk} + y_{ki})$.

Let $e_1, e_2 \in \delta(S) \setminus \delta(i)$ and $x' = \sum_{e \in (E(S) \cup E(V \setminus S)) \setminus \delta(i)} \chi_e$. Consider the solution $(x' + \chi_{e_1} + \chi_{e_2}, \gamma_{ij0})$. Clearly this solution is in $\mathcal{F}$. Let $e_3 \in \delta(S) \setminus (\delta(i) \cup \{e_1, e_2\})$. It can be seen that $(x' + \chi_{e_1} + \chi_{e_3}, \gamma_{ij0})$ and $(x' + \chi_{e_2} + \chi_{e_3}, \gamma_{ij0})$ are also solutions in $\mathcal{F}$. This shows that $a_e = \sigma$ for some $\sigma \in \mathbb{R}$ for all $e \in \delta(S) \setminus \delta(i)$. Now let $e_1, e_2, e_3 \in \delta(S) \setminus \delta(j)$ and $x^{''} = \sum_{e \in (E(S) \cup E(V \setminus S)) \setminus \delta(j)} \chi_e$. Consider the similar solutions $(x^{''} + \chi_{e_1} + \chi_{e_2}, \gamma_{jk0})$, $(x^{''} + \chi_{e_1} + \chi_{e_3}, \gamma_{jk0})$, and $(x^{''} + \chi_{e_2} + \chi_{e_3}, \gamma_{jk0})$. Since all three solutions are in $\mathcal{F}$, we can conclude that $a_e = \sigma$ for all $e \in \delta(S)$.

To find the coefficients of the edges in $E(S)$, let $\{u, v\} \in E(S) \setminus \delta(i)$ and $e_1, e_2 \in \delta(S) \setminus \delta(i)$ such that $u \in e_1$ and $v \in e_2$. We can see that $(x' + \chi_{e_1} + \chi_{e_2}, \gamma_{ij0})$ is a solution in $\mathcal{F}$. As $(x' + \chi_{e_1} + \chi_{e_2} - \chi_{uv}, \gamma_{ij0})$ is also in $\mathcal{F}$, $a_e = 0$ for all $e \in E(S) \setminus \delta(i)$. In addition, we can easily extend this result. Let $\{i, u\} \in E(S) \setminus \delta(j)$, $e_1, e_2 \in \delta(S) \setminus \delta(j)$ such that $i \in e_1$ and $u \in e_2$. Since $(x^{''} + \chi_{e_1} + \chi_{e_2}, \gamma_{jk0})$ and $(x^{''} + \chi_{e_1} + \chi_{e_2} - \chi_{iu}, \gamma_{jk0})$ are both in $\mathcal{F}$, we can say that $a_e = 0$ for all $e \in E(S) \setminus \{\{i, j\}\}$. For the remaining edge $\{i, j\}$, let $e_1, e_2 \in \delta(S) \setminus \delta(k)$ such that $i \in e_1$ and $j \in e_2$. We also define $x^{'''} = \sum_{e \in (E(S) \cup E(V \setminus S)) \setminus \delta(k)} \chi_e$. It can be easily seen that $(x^{'''} + \chi_{e_1} + \chi_{e_2}, \gamma_{ki0})$ and $x^{'''} + \chi_{e_1} + \chi_{e_2} - \chi_{ij}, \gamma_{ki0})$ are both in $\mathcal{F}$. Therefore, $a_e = 0$ for all $e \in E(S)$.

The case for the edges in $V \setminus S$ is simpler. If $|V \setminus S| = 1$ there is no edge in $V \setminus S$, so we assume $|V \setminus S| \geq 3$. Let $\{u, v\} \in E(V \setminus S)$, $e_1, e_2 \in \delta(S)$ such that $u \in e_1$ and $v \in e_2$. Clearly, $(x' + \chi_{e_1} + \chi_{e_2}, \gamma_{ij0})$ and $(x' + \chi_{e_1} + \chi_{e_2} - \chi_{uv}, \gamma_{ij0})$ are both solutions in $\mathcal{F}$. So we conclude that, $a_e = 0$ for all $e \in E(V \setminus S)$.

Let $l \in V \setminus \{0, i, j, k\}$ and $u, v, w \in V \setminus \{l, i\}$. Let e_1, e_2 be two edges in $\delta(S) \setminus \delta(i)$. Defining $\overline{x} = \sum_{e \in (E(S) \cup E(V \setminus S)) \setminus (\delta(i) \cup \delta(l))} \chi_e$ we can find the solution $(\overline{x} + \chi_{e_1} + \chi_{e_2}, \gamma_{ij0} + \gamma_{luv})$ in $\mathcal{F}$. Observe that the solution $(\overline{x} + \chi_{e_1} + \chi_{e_2}, \gamma_{ij0} + \gamma_{lvw})$ is also in $\mathcal{F}$. Moreover, $(x'' + \chi_{e_1} + \chi_{e_2}, \gamma_{jk0})$ is in $\mathcal{F}$. So we have $b_{uv} = 0$ for all $(u, v) \in A$ such that $u \in V \setminus \{0, i, j, k\}$.

Similarly, let $u, v, w \in V \setminus \{i, j\}$ and define $\hat{x} = \sum_{e \in (E(S) \cup E(V \setminus S)) \setminus (\delta(i) \cup \delta(j))} \chi_e$. Clearly, $(\hat{x} + \chi_{e_1} + \chi_{e_2}, \gamma_{iuv} + \gamma_{jk0})$ is a solution in $\mathcal{F}$ where $e_1, e_2 \in \delta(S) \setminus (\delta(i) \cup \delta(j))$. Observe that the solution $(\hat{x} + \chi_{e_1} + \chi_{e_2}, \gamma_{ivw} + \gamma_{jk0})$ is also in $\mathcal{F}$. Moreover, $(x'' + \chi_{e_1} + \chi_{e_2}, \gamma_{jk0})$ is also a solution in $\mathcal{F}$. These solutions can be constructed for nodes j and k in a similar way. So we have $b_{uv} = 0$ for all $(u, v) \in A \setminus \{(i, j), (j, k), (k, i)\}$.

Finally, let $e_1, e_2 \in \delta(S) \setminus (\delta(i) \cup \delta(j) \cup \delta(k))$. Clearly, the solutions $(x' + \chi_{e_1} + \chi_{e_2}, \gamma_{ij0})$, $(x'' + \chi_{e_1} + \chi_{e_2}, \gamma_{jk0})$, and $(x''' + \chi_{e_1} + \chi_{e_2}, \gamma_{ki0})$ are all in $\mathcal{F}$. Considering these solutions together with the solution $(\sum_{e \in E(V \setminus S)} \chi_e, \sum_{u \in S} \gamma_{u0v})$ where $v \in V \setminus (S \cup \{0\})$, we can see that $b_a = -2\sigma$ for $a \in \{(i, j), (j, k), (k, i)\}$.

Therefore, $ax + by = \beta$ is a multiple of $x(\delta(S)) - 2(y_{ij} + y_{jk} + y_{ki}) = 0$ and $\mathcal{F}$ is a facet of $\mathcal{P}$. □

In Section 5.2, we derived two families of projection inequalities. These inequalities involve the variables t_i's. Here we first rewrite these inequalities without using the t_i variables and then investigate under which conditions they define facets.

Eliminating the variable t_i's in the projection inequalities (5.24) and (5.25), we obtain

$$x(\delta(S)) + \sum_{k\in V\setminus\{i,j\}} y_{ik} - y_{ij} \geq 2. \tag{5.36}$$

and

$$x(\delta(S)) + 2\sum_{l\in V\setminus(S\cup\{j\})} y_{il} \geq 2 \tag{5.37}$$

Theorem 5.11 *Let $S \subseteq V \setminus \{0\}$ such that $S \neq \emptyset$, $i \in S$, and $j \in S \setminus \{i\}$. If $|S| \geq 3$ and $|V \setminus S| \geq 3$ then inequality (5.36) defines a facet of $\mathcal{P}$.*

Proof Let $S \subseteq V \setminus \{0\}$ such that $S \neq \emptyset$, $i \in S$, $j \in S \setminus \{i\}$, and $\mathcal{F} = \{(x, y) \in \mathcal{P} : x(\delta(S)) + \sum_{k\in V\setminus\{i,j\}} y_{ik} - y_{ij} = 2\}$. Suppose that $|S| \geq 3$, $|V \setminus S| \geq 3$, and every solution (x, y) in $\mathcal{F}$ also satisfies $ax + by = \beta$.

It can be seen that $a_e = \sigma$ for some $\sigma \in \mathbb{R}$ for every $e \in \delta(S)$ and $b_{ik} = \sigma$ for every $(i, k) \in A$ such that $k \in V \setminus S$. Similarly, $a_e = 0$ for every $e \in E \setminus \delta(S)$ and $b_{uv} = 0$ for every $(u, v) \in A$ such that $u \neq i$. The details of how these coefficients are found are provided in the proof of Theorem 5.9.

Let $u \in V \setminus S$, $k \in S \setminus \{i, j\}$, and $e_1, e_2 \in \delta(S) \setminus \delta(i)$. We define $x' = \sum_{e\in E(V\setminus S)\cup E(S)\setminus\delta(i)} \chi_e + \chi_{e_1} + \chi_{e_2}$. Then (x', γ_{iju}) and (x', γ_{ijk}) are both solutions in $\mathcal{F}$ showing that $b_{ik} = \sigma$ for all $k \in V \setminus \{i, j\}$.

Finally, consider the solutions (x', γ_{ijk}) and (x'', y'') where $x'' = \sum_{e\in E(V\setminus S)} \chi_e$, $y'' = \sum_{l\in S} \gamma_{luv}$, and $u, v \in V \setminus S$. Clearly both solutions are in $\mathcal{F}$. Since $a_e = \sigma$ for $e \in \delta(S)$, $b_{ik} = \sigma$ for $k \in V \setminus \{i, j\}$ and the other coefficients are all zero we have $b_{ij} = -\sigma$.

Thus, $ax + by = \beta$ is a multiple of $x(\delta(S)) + \sum_{k\in V\setminus\{i,j\}} y_{ik} - y_{ij} = 2$ and $\mathcal{F}$ is a facet of $\mathcal{P}$. $\square$

Theorem 5.12 *Let $S \subseteq V \setminus \{0\}$ such that $S \neq \emptyset$, $i \in S$, and $j \in V \setminus S$. If $|S| \geq 4$ and $|V \setminus S| \geq 3$ then inequality (5.37) defines a facet of $\mathcal{P}$.*

Proof Suppose that $|S| \geq 4$ and $|V \setminus S| \geq 3$. Let $\mathcal{F} = \{(x,y) \in \mathcal{P} : x(\delta(S)) + 2\sum_{l\in V\setminus(S\cup\{j\}} y_{il} = 2\}$. Suppose that every solution (x,y) in $\mathcal{F}$ also satisfies $ax + by = \beta$. We will show that $ax+by = \beta$ is a multiple of $x(\delta(S))+2\sum_{l\in V\setminus(S\cup\{j\}} y_{il} = 2\}$.

It can be shown that $a_e = 0$ for all $e \in E \setminus \delta(S)$, and $a_e = \sigma$ for some $\sigma \in \mathbb{R}$ for all $e \in \delta(S)$. We can also show that $b_{il} = 0$ for all $l \in S \setminus \{i\}$, and $b_{kl} = 0$ for all $(k,l) \in A$ such that $k \neq i$. For the calculations of these coefficients, one can refer to the proof of Theorem 5.9.

Let e_1, e_2 be two edges in $\delta(S)\setminus\delta(i)$ and $k_1, k_2 \in S\setminus\{i\}$. Consider the solution $(x, \gamma_{ik_1k_2})$ where $x = \sum_{e\in E(S)\cup E(V\setminus S)\setminus\delta(i)} \chi_e + \chi_{e_1} + \chi_{e_2}$. This solution and (x, γ_{ijk_1}) are both in $\mathcal{F}$. So we have $b_{ik_2} = b_{ij} = 0$.

Finally, let $k \in V\setminus(S\cup\{j\})$. We define $\overline{y} = \sum_{l\in S} \gamma_{ljk}$. Note that (x, γ_{ijk_1}) is a solution in $\mathcal{F}$. In addition $(\sum_{e\in E(V\setminus S)} \chi_e, \overline{y})$ is also in $\mathcal{F}$. As $a_e = 0$ for $e \in E(S)$, $a_e = \sigma$ for $e \in \delta(S)$, $b_{uv} = 0$ for $(u,v) \in A$ with $u \neq i$ and $b_{ik_1} = 0$ we can see that $b_{ik} = 2\sigma$ for every $k \in V \setminus (S \cup \{j\})$.

Therefore $ax + by = \beta$ is a multiple of $x(\delta(S)) + 2\sum_{l\in V\setminus(S\cup\{j\})} y_{il} = 2$ and $\mathcal{F}$ is a facet of $\mathcal{P}$. $\square$

5.3.4 Extended F-partition inequalities

We can also extend the family of F-partition inequalities to the dual homing problem. Let $V_0, \ldots, V_p$ be a partition of V such that $V_l \neq \emptyset$, for $l = 0, \ldots, p$ and $0 \in V_0$. Let $i_l \in V_l$ be a fixed node for $l = 1, \ldots, p$ and $F \subseteq \delta(V_0)$ such that $|F| = 2k+1$ for some $k \geq 0$ and integer. Let $\delta(V_0, \ldots, V_p)$ be the set of edges whose endpoints are in different sets of the partition.

We can obtain an F-partition inequality as follows. Consider the following inequalities which are valid for the dual homing problem.

$$
\begin{aligned}
x(\delta(V_l)) + \sum_{j \in V \setminus V_l} y_{i_l j} &\geq 2 && l = 1, \ldots, p \\
-x_e &\geq -1 && \forall e \in F \\
x_e &\geq 0 && \forall e \in \delta(V_0) \setminus F.
\end{aligned}
$$

Adding up these inequalities and dividing the resulting inequality by 2 yields

$$x(\delta(V_0, \ldots, V_p) \setminus F) + \frac{\sum_{l=1}^{p} \sum_{j \in V \setminus V_l} y_{i_l j}}{2} \geq p - \frac{|F|}{2} \tag{5.38}$$

Observe that the left hand side of the inequality may be fractional which will prevent rounding the right hand side up. So we need to show that the right hand side can be rounded up even if the left hand side is fractional.

Theorem 5.13 *Let $|F| = 2k + 1$ for some $k \geq 0$ integer. Then*

$$x(\delta(V_0, \ldots, V_p) \setminus F) + \frac{\sum_{l=1}^{p} \sum_{j \in V \setminus \{i_l\}} y_{i_l j}}{2} \geq p - k \tag{5.39}$$

is valid for $\mathcal{P}$.

Proof If the left hand side of inequality (5.39) is integer, then it is valid since inequality (5.38) is a valid inequality. So we may assume $\sum_{l=1}^{p} \sum_{j \in V \setminus \{i_l\}} y_{i_l j}$ to be odd to have the left hand side fractional. This implies that there exists at least one fixed node i_l for some $l = 1, \ldots, p$, which is assigned to one hub in V_l and to one hub in $V \setminus V_l$. In this case there exists at least one node, say j, in V_l other than i_l to which i_l is assigned. Since there is a cut inequality induced by j and V_l, we have $x(\delta(V_l)) \geq 2$. Considering the inequalities used to obtain the extended F-partition inequality it can be seen that the cut inequality defined by i_l and V_l contributes to the left hand side 1.5 while contributing to the right hand side only 1, resulting in a 0.5 surplus for the left hand side which is at least as large as the increase in the right hand side after rounding up. Therefore, the right hand side of the extended F-partition inequality can be rounded up when

$|F|$ is odd even if the left hand side is not integer, and hence inequality (5.39) is valid for $\mathcal{P}$. $\square$

We now give sufficient conditions for these inequalities to be facet defining for $\mathcal{P}$.

Theorem 5.14 *Inequality (5.39) defines a facet for $\mathcal{P}$ if*

- *G_l is 3-edge connected for $l = 0, \ldots, p$,*
- *$|F \cap \delta(V_l)| \leq 1$ and $F \cap \delta(j) = \emptyset$ for $l = 1, \ldots, p$ and $j \in V_l \setminus \{i_l\}$,*
- *$|F \cap \delta(j)| \leq 1$ for $j \in V_0 \setminus \{0\}$.*

Proof Let $\mathcal{F} = \{(x, y) \in \mathcal{P} : x(\delta(V_0, \ldots, V_p) \setminus F) + \frac{\sum_{l=1}^{p} \sum_{j \in V \setminus \{i_l\}} y_{i_l j}}{2} = p - k\}$. Assume that every solution $(x, y) \in \mathcal{F}$ also satisfies $ax + by = \beta$. Without loss of generality, assume that $\delta(i_l) \cap F \neq \emptyset$ for $l = 1, \ldots, 2k+1$. Clearly, $p \geq 2k+1$.

Let $E_1 = (\cup_{l=0}^{p} E(V_l)) \cup \{i_1, i_2\} \cup \{i_2, i_{2k+2}\} \cup_{l=2k+2}^{p-1} \{i_l, i_{l+1}\} \cup \{i_p, i_3\} \cup_{l=2}^{k} \{i_{2l}, i_{2l+1}\} \cup F$ and $x = \sum_{e \in E_1} \chi_e$. Note that in E_1 there are some edges which exist only if $p > 2k+1$. So if $p = 2k+1$, we need to define $E_1 = (\cup_{l=0}^{p} E(V_l)) \cup \{i_1, i_2\} \cup \{i_2, i_3\} \cup_{l=2}^{k} \{i_{2l}, i_{2l+1}\} \cup F$. Then the solution $(x, 0)$ is in $\mathcal{F}$. The solution $(x - \chi_e, 0)$ is also in $\mathcal{F}$ for every $e \in E(V_l)$ for $l = 0, \ldots, p$ since $G(V_l)$ is 3-edge connected. Therefore $a_e = 0$ for all $e \in E(V_l)$ for $l = 0, \ldots, p$. Similarly, $(x - \chi_e, 0)$ is in $\mathcal{F}$ for $e = \delta(i_2) \cap F$. By symmetry, we can show that $a_e = 0$ for every $e \in F$.

For simplicity, we use i_0 to represent node 0. Let $l \in \{0, \ldots, p\}$, $j \in V_l \setminus \{i_l\}$, and $u, v, w \in V_l \setminus \{j\}$. Since the solutions $(x - \sum_{e \in \delta(j)} x_e \chi_e, \gamma_{juv})$ and $(x - \sum_{e \in \delta(j)} x_e \chi_e, \gamma_{juw})$ are both in $\mathcal{F}$ we have $b_{ju} = \xi_j$ for some $\xi_j \in \mathbb{R}$ for $j \in V_l \setminus \{i_l\}$ and $u \in V_l \setminus \{j\}$. Comparing the solution $(x - \sum_{e \in \delta(j)} x_e \chi_e, \gamma_{juv})$ with $(x, 0)$ we can show that $\xi_j = 0$ for $j \in V_l \setminus \{i_l\}$ as $a_e = 0$ for all $e \in E(V_l)$.

Let $e \in \delta(V_1) \setminus F$. Observe that the solution $(x - \chi_{i_1 i_2} + \chi_e, 0)$ is in $\mathcal{F}$. So $a_e = \alpha_1$ for all $e \in \delta(V_1) \setminus F$ and for some $\alpha_1 \in \mathbb{R}$. By symmetry, we can show that $a_e = \alpha_l$ for $e \in \delta(V_l) \setminus F$ for $l = 1, \ldots, 2k+1$.

Now as $(\delta(V_j) \setminus F) \cap (\delta(V_l) \setminus F) \neq \emptyset$ for any j and l such that $1 \leq j < l \leq 2k+1$, we have $\alpha_j = \alpha_l$. So we can conclude that $a_e = \alpha$ for some $\alpha \in \mathbb{R}$ for $e \in \delta(V_0, \ldots, V_{2k+1}) \setminus F$.

Let $l \in \{0, \ldots, 2k+1\} \setminus \{3\}$ and $e \in (\delta(V_{2k+2}) \setminus E_1) \cap \delta(V_l)$. The solution $(x - \chi_{i_2 i_{2k+2}} + \chi_e, 0)$ is in $\mathcal{F}$, and hence $a_e = \alpha$. Now by changing the roles of V_3 and V_1, we can also show that $a_e = \alpha$ for all $e \in [V_{2k+2}, V_3]$. By symmetry, we can conclude that $a_e = \alpha$ for all $e \in [V_i, V_j]$ for $i = 0, \ldots, 2k+1$ and $j = 2k+2, \ldots, p$. Note that this paragraph is necessary for the case $p > 2k+1$ as there is no such an edge e if $p = 2k+1$.

Let $u, v, w \in V \setminus V_1$, $e' \in F \cap \delta(i_1)$, and we define $\overline{x} = x - \sum_{e \in \delta(V_1)} x_e \chi_e - \sum_{e \in E(V_1)} \chi_e$ and $\overline{y} = \sum_{k \in V_1 \setminus \{i_1\}} \gamma_{kuv}$. Clearly, $(\overline{x}, \overline{y} + \gamma_{i_1 uv})$ and $(\overline{x}, \overline{y} + \gamma_{i_1 uw})$ are both solutions in $\mathcal{F}$. Therefore, we have $b_{i_1 u} = \theta_1$ for $u \in V \setminus V_1$. Moreover, $(\overline{x} + \chi_{i_1 i_2} + \chi_{e'}, \overline{y})$ is also in $\mathcal{F}$. As $a_{e'} = 0$, $a_{i_1 i_2} = \alpha$, and $b_{i_1 u} = b_{i_1 v} = \theta_1$ we obtain $b_{i_1 u} = \alpha/2$. By symmetry, we can extend this results to $b_{i_l u} = \alpha/2$ for $l = 1, \ldots, 2k+1$ and $u \in V \setminus V_l$.

The rest of the proof computes the coefficients of the variables that appear only in the case $p > 2k+1$. So the following three paragraphs can be omitted if $p = 2k+1$.

Let $u, v, w \in V \setminus V_{2k+2}$. Let $x' = x - \sum_{e \in E(V_{2k+2})} \chi_e - \chi_{i_2 i_{2k+2}} - \chi_{i_{2k+2} i_{2k+3}} + \chi_{i_2 i_{2k+3}}$ and $y' = \sum_{k \in V_{2k+2} \setminus \{i_{2k+2}\}} \gamma_{kuv}$. It can be seen that the solutions $(x', y' + \gamma_{i_{2k+2} uv})$ and $(x', y' + \gamma_{i_{2k+2} uw})$ are both in $\mathcal{F}$. This implies that $b_{i_{2k+2} u} = \theta_{2k+2}$ for some $\theta_{2k+2} \in \mathbb{R}$ for all $u \in V \setminus \{i_{2k+2}\}$ and by symmetry, we can conclude that $b_{i_l m} = \beta_l$ for all $l = 2k+2, \ldots, p$ and $u \in V \setminus V_l$.

Let $E_2 = (\cup_{l=0}^{2k+1} E(V_l)) \cup \{i_1, i_2\} \cup (\cup_{l=1}^{k} \{i_{2l}, i_{2l+1}\}) \cup F$ and $u, v \in V_0$. We can define $x'' = \sum_{e \in E_2} \chi_e$ and $y'' = \sum_{l=2k+2}^{p} \sum_{j \in V_l} \gamma_{juv}$. Observe that (x'', y'') is a solution in $\mathcal{F}$. We can also construct a solution in $\mathcal{F}$ as $(x'' - \chi_{i_2 i_3} + \chi_{e_1} + \chi_{e_2} + \sum_{e \in E(V_{2k+2})} \chi_e, y^{''} - \sum_{j \in V_{2k+2}} \gamma_{juv})$ $\mathcal{F}$, where $e_1 \in [V_2, V_{2k+2}]$ and $e_2 \in [V_3, V_{2k+2}]$. Since $a_{i_2 i_3} = a_{e_1} = a_{e_2} = \alpha$, we can see $\beta_{2k+2} = \alpha/2$. By symmetry, we have $\beta_l = \alpha/2$ for $l = 2k+2, \ldots, p$.

Now considering $(x, 0)$ and (x', y') together reveals that $a_e = \alpha$ for $e \in [V_j, V_l]$ with j and l in $\{2k + 2, \ldots, p\}$.

Therefore, we can conclude that $ax+by = \beta$ is a multiple of inequality (5.39).$\square$

5.4 Separation Algorithms

The mathematical model we developed has exponential number of constraints. So we need to develop a branch-and-cut algorithm to solve this model to optimality as we did for the 2ECSSP. In this section we describe the separation algorithms used to identify the violated inequalities utilized in the branch-and-cut algorithm. The separation problems are quite similar to the ones described in Section 4.2. However, for the sake of completeness, we provide detailed descriptions of the separation algorithms.

For a given fractional solution (x^*, y^*) we define the following sets $V_h = \{i \in V : \sum_{j \in V \setminus \{i\}} y^*_{ij} = 0\} \cup \{0\}, V_{ph} = \{i \in V : \sum_{j \in V \setminus \{i\}} y^*_{ij} > 0 \text{ and } x^*(\delta(i)) > 0\}, V_u = \{i \in V : x^*(\delta(i)) = 0 \text{ and } \exists j, k \in V \setminus \{i\} \text{ such that } y^*_{ij} = y^*_{ik} = 1\}$ and $V_{pu} = \{i \in V \setminus V_u : x^*(\delta(i)) = 0\}$. These sets form a partition of V and their elements will be called *hubs*, *partial hubs*, *users*, and *partial users*, respectively. The nodes of the backbone network are composed of the hubs and partial hubs. We define $V^* = V_h \cup V_{ph}$, $E^* = \{\{i, j\} \in E : x^*_{ij} > 0\}$ and $A^* = \{(i, j) \in A : 0 < y^*_{ij} < 1\}$. $G^* = (V^*, E^*)$ is our support graph. The support graph may be disconnected and we use $G^i = (V^i, E^i)$ for $i = 0, \ldots, r$ be the i^{th} connected component of G^*. Without loss of generality, we assume $0 \in V^0$. Clearly $G^0 = G^*$ if G^* is connected.

Violated relation inequalities (5.26) are found by complete enumeration. The cut (5.29), double cut (5.37) and small cut inequalities (5.36) can be separated in polynomial time by solving minimim cut problems. For all three inequalities the separation is performed in multiple phases one of which is a heuristic phase. For the heuristic phase Hao-Orlin algorithm is used. This algorithm finds $n - 1$ minimum cuts where n is the number of nodes on the graph. Therefore $n - 1$

different cut sets are obtained at the end of the algorithm. The details of how these cut sets are used for separation are described in the following sections. We use this algorithm on G^0, and we set the capacity of edge $e \in E^0$ to x_e^*. The root node 0 is chosen as the initial sink for the algorithm so that we ensure 0 is always in $V \setminus S$. As we ignore the assignment part of the fractional solution this part constitutes only a heuristic method.

In all separation problems, we use the Hao-Orlin algorithm to find the global minimum cut and the Goldberg-Tarjan algorithm [17] is used to solve the minimum cut problems. Now we can present the details of the separation algorithms.

5.4.1 Cut inequalities

A cut inequality (5.29) is defined by a node set $S \subseteq V \setminus \{0\}$ and a fixed node $i \in S$. We can check if there is a violated cut inequality with a fixed node i by solving a minimum cut problem [30]. So we can find if there is a violated cut inequality by solving $O(|V|)$ minimum cut problems. We perform this analysis in three phases. In the first phase, we use the information on the connectivity of the support graph. It can be seen that the each connected component of the support graph that does not include the root node 0 yields a violated cut inequality. Let the support graph be unconnected. Then for a given connected component $G^j = (V^j, E^j)$ with $j > 0$ and a fixed node $i \in V^i$ we define $S_i = V^j \cup \{k \in V_u \cup V_{pu} : y_{ik}^* > 0\}$. As $x^*(\delta(V^j)) = 0$ and $t_i^* > 0$ the cut inequality defined by S_i and i is violated. So we can generate a violated cut inequality for every $i \in V^* \setminus V^0$. Note that we do not consider nodes in V^0 as fixed nodes since V^0 is the node set of the connected component that includes the root node.

In the second phase, we use the cut sets obtained from the Hao-Orlin algorithm. Let $S_1, \ldots, S_{|V^0|-1}$ be the cut sets. If the capacity of the cut is greater than or equal to 2 then there is no violated cut inequality associated with this cut. However, if the capacity is less than 2, we need to check if there is a violated cut inequality defined by this cut set by taking y^* values into account. Therefore, for $i = 1, \ldots, |V^0| - 1$ such that the capacity of $[S_i, V^0 \setminus S_i]$ is less than

2, we calculate the violation of the cut inequality defined by S_i and j for every $j \in S_i \cup V_u \cup V_{ph}$. If there is at least one violated cut inequality, we choose the one with the maximum violation. We break the ties arbitrarily. Note that if the fixed node, say $j \in V_u \cup V_{pu}$, is chosen for a given cutset S, the cutset must be extended as $S \cup \{j\}$ since $j \notin S$.

Finally, we use exact separation. Let $G_i^* = (V^* \cup \{i\}, E_i^*)$ where $E_i^* = E^* \cup \{\{i,j\} : j \in V^* \text{ and } (i,j) \in A^*\}$. Let $u, v \in V \setminus \{i\}$. Then we set the capacity of edge $\{u,v\}$ to x_{uv}^* and the one of edge $\{i,v\}$ to $x_{iv}^* + y_{iv}^*$. By solving a minimum cut problem between nodes i and 0 we can determine if there is a violated cut inequality with fixed node i. If the minimum cut capacity is less than 2, then there is a violated cut inequality.

Note that we do not include the user and partial user nodes in the graph that we use for the minimum cut computation unless they are not the fixed node. This is because there is no adjacent edge to them in the support graph. These nodes are put together with node 0. However, if the fixed node of a given cut inequality is assigned to a user or partial user node j then we move it from $V \setminus S$ to S. This operation does not affect the first term of the cut inequality but reduces the second term and hence the violation increases. Although nodes cannot be assigned to users in feasible solutions there may be such assignments in the fractional solutions since the subproblems do not include all relation inequalities.

As finding a minimum cut between i and 0 provides the most violated cut inequality with fixed node i, we need to solve minimum cut problems for every node of the backbone network except for the root node. This is necessary as the cut inequalities are in the model formulation and must be separated exactly. However, we can use the information obtained from the heuristic step to eliminate some nodes from consideration. We do not solve a minimum cut problem for a node i if a violated cut inequality that uses i as the fixed node is found in the heuristic phase. So we define $C = V^0 \cup V_{pu} \setminus \{0\}$ as the set of candidate nodes and at each step of the heuristic we remove the fixed node in case a violated cut inequality is identified. Note that the nodes of V_u are excluded from this set due to the following reasons. Let $i \in V_u$, $y_{ij}^* = y_{ik}^* = 1$ and $S \subseteq V \setminus \{0\}$ be such that

$i \in S$. Suppose that i and S define a violated cut inequality. Clearly at least one of j, k must be in S, otherwise the inequality is not violated. If $j \in S, k \notin S$, ($j, k \in S$). This implies $\sum_{j \in V \setminus S} y^*_{il} = 1$, ($\sum_{j \in V \setminus S} y^*_{il} = 0$) and $x^*(\delta(S)) < 1$, ($x^*(\delta(S)) < 2$). Clearly, the pair j and S also induce a violated cut inequality. For this reason, we do not consider the nodes of V_u in the third phase. In addition we do not include the nodes of $V^* \setminus V^0$ in C as we add cut inequalities for them in the first phase.

If at least one cut with capacity less than 2 is found in the heuristic phase and $C \neq \emptyset$ we pass to the last step, the exact separation phase, in which we solve a minimum cut problem between i and 0 on G^*_i for every $i \in C$.

Our separation procedure is given in Algorithm 4.

5.4.2 Double-cut inequalities

A double-cut inequality (5.37) is defined by a node set $S \subseteq V \setminus \{0\}$ and two fixed nodes $i \in S$ and $j \in V \setminus S$. We can determine if there is a violated double-cut inequality with fixed nodes i, j by solving a minimum cut problem between nodes i and 0. If the minimum cut capacity is less than 2, this means there is a violation. Therefore double-cut inequalities can be separated exactly in polynomial time by solving $O(|V|^2)$ minimum cut problems.

We start with considering the cut sets, $S_1, \ldots, S_{|V^0|-1}$, obtained in the heuristic phase. For a given cut set S with capacity less than 2, the node pair $i \in S$ and $j \in V \setminus S$ that minimizes $\sum_{k \in (V_h \cup V_{ph}) \setminus (S \cup \{j\})} y^*_{ik}$ is found. Note that the nodes in $V_u \cup V_{pu}$ are not included in this quantity as they can be moved into S without affecting $x^*(\delta(S))$. If there is at least one violated double-cut inequality, we choose the one with the maximum violation and break the ties arbitrarily. Note that the cutset S must be extended as $S \cup \{i\}$ if $i \in V_u \cup V_{pu}$ since $i \notin S$.

Let $G^*_{ij} = (V^* \cup \{i\}, E^*_{ij})$ where $E^*_{ij} = E^* \cup \{\{i, k\} : k \in V^* \setminus \{j\}$ and $(i, k) \in A^*\} \cup \{\{j, 0\}\}$. Let $v \in V \setminus \{i\}$. Then we set the capacity of edge $e = \{u, v\}$ to x^*_e if $i \notin e$ or $j \in e$, and to $x^*_{iv} + 2y^*_{iv}$. Note that node j must be in $V \setminus S$ to define

Algorithm 4: Cut Inequality Separation

```
Input: $(x^*, y^*), G^i = (V^i, E^i)$ for $i = 0, \dots, r$, $G_i^* = (V^* \cup \{i\}, E_i^*)$ for
       $i \in V \setminus \{0\}$
1  begin
2      $C \leftarrow V_{pu} \cup V^0 \setminus \{0\}$
3      if $r > 0$ then
4          for $i = 1$ to $r$ do
5              forall $k \in V^i \cup V_u \cup V_{pu}$ do $z_k = 2 - \sum_{j \in V^* \setminus V^i} y_{kj}^*$
6              $j \leftarrow argmax_{k \in V^i \cup V_u \cup V_{pu}} \{z_k\}$
7              if $z_j > 0$ then
8                  $S = V^i \cup \{k \in V_u \cup V_{pu} : y_{jk}^* > 0\}$ and $j$ form a violated cut
                   ineq.
9                  $C \leftarrow C \setminus \{j\}$
10     Use Hao-Orlin algorithm on $G^0$ and find $|V^0| - 1$ cutsets denoted by
       $S_1, \dots, S_{|V^0|-1}$
11     for $l = 1$ to $|V^0| - 1$ do
12         if capacity of $[S_l, V^0 \setminus S_l]$ is less than $2$ then
13             forall $i \in S_l \cup V_u \cup V_{pu}$ do $z_i = 2 - x^*(\delta(S_l)) - \sum_{j \in V^* \setminus S_l} y_{ij}^*$
14             $j \leftarrow argmax_{i \in S_l \cup V_u \cup V_{pu}} \{z_i\}$
15             if $z_j > 0$ then
16                 $S_l \cup \{k \in V_u \cup V_{pu} : y_{jk}^* > 0\}$ and $j$ form a violated cut
                   inequality
17                 $C \leftarrow C \setminus \{j\}$
18     forall $i \in C$ do
19         Find minimum cut $[S, (V^0 \cup \{i\}) \setminus S]$ between $i$ and $0$ on $G_i^*$
20         if capacity of $[S, (V^0 \cup \{i\}) \setminus S]$ is less than $2$ then
               $S \cup \{j \in V_u \cup V_{pu} : y_{ij}^* > 0\}$ and $i$ form a violated cut inequality
21 end
```

the inequality. Therefore we add the last edge between 0 and j with capacity 2. Solving a minimum cut problem between i and 0 on G^*_{ij} will show if there is a violated double-cut inequality with these fixed nodes.

Similar to the cut inequality separation, we do not use the nodes that has been used as the fixed node i in the double-cut inequalities added in the heuristic phase. We provide the separation procedure in Algorithm 5.

Algorithm 5: Double-cut Inequality Separation

```
Input: $(x^*, y^*)$, $G^0 = (V^0, E^0)$, $G^*_{ij} = (V^* \cup \{i\}, E^*_{ij})$ for
       $i \in V \setminus \{0\}, j \in V \setminus \{i\}$
1  begin
2      $C \leftarrow V_{pu} \cup V^0 \setminus \{0\}$
3      Use Hao-Orlin algorithm on $G^0$ and find $|V^0| - 1$ cutsets denoted by
       $S_1, \dots, S_{|V^0|-1}$
4      for $l = 1$ to $|V^0| - 1$ do
5          if capacity of $[S_l, V^0 \setminus S_l]$ is less than 2 then
6              $B \leftarrow \{(i,j) : i \in S_l \cup V_u \cup V_{pu}, j \in V \setminus (S_l \cup V_u \cup V_{pu})\}$
7              forall $(i,j) \in B$ do $z_{ij} = 2 - x^*(\delta(S_l)) - 2\sum_{l \in V^0 \setminus (S_l \cup \{j\})} y^*_{il}$
8              $(u,v) \leftarrow argmax_{(i,j) \in B}\{z_{ij}\}$
9              if $z_{uv} > 0$ then
10                 $S \cup \{j \in V_u \cup V_{pu} : y^*_{uj} > 0\}$ and $u, v$ form a violated
                   double-cut inequality
11                 $C \leftarrow C \setminus \{u\}$
12     forall $i \in C$ do
13         forall $j \in V_h \cup V_{ph} \setminus \{i\}$ do
14             Find minimum cut $[S, (V^0 \cup \{i\}) \setminus S]$ between $i$ and 0 on $G^*_{ij}$
15             if capacity of $[S, (V^0 \cup \{i\}) \setminus S]$ is less than 2 then
               $S \cup \{k \in V_u \cup V_{pu} : y^*_{ik} > 0\}$ and $i, j$ form a violated double-cut
               inequality
16 end
```

5.4.3 Small-cut inequalities

Like double-cut inequalities, small-cut inequalities (5.36) are also defined by two fixed nodes $i \in S$ and $j \in S \setminus \{i\}$, and a node set $S \subseteq V \setminus \{0\}$. We first analyze the cut sets,$S_1, \dots, S_{|V^0|-1}$, obtained by the Hao-Orlin algorithm. If the capacity of a given cut set S is less than 2, we look for the node pair $i, j \in S$ that maximizes $t_i^* + y_{ij}^*$.

For the nodes which are not used to define violated cut inequalities in the heuristic phase we apply exact separation. Let $G_{ij}^* = (V^* \cup \{i\}, E_{ij}^* \cup \{\{i,j\}\})$. We set the capacity of edge e to x_e^* and capacity of $\{i,j\}$ to 2 so that i and j are in the same set. We solve the minimum cut problem between i and 0 on G_{ij}^* and if the capacity of the minimum cut is less than $2(t_i^* + y_{ij}^*)$, then a violated small-cut inequality is found. By this way, small-cut inequalities can be separated exactly by solving $O(|V|^2)$ minimum cut problems.

The separation procedure is presented in Algorithm 6.

Algorithm 6: Small-cut Inequality Separation

```
   Input: $(x^*, y^*)$, $G_{ij}^* = (V^* \cup \{i,j\}, E_{ij}^* \cup \{\{i,j\}\})$ for
          $i \in V \setminus \{0\}, j \in V \setminus \{i\}$
 1 begin
 2     $C \leftarrow V_{pu} \cup V^0 \setminus \{0\}$
 3     Use Hao-Orlin algorithm on $G^0$ and find $|V^0| - 1$ cutsets denoted by
       $S_1, \ldots, S_{|V^0|-1}$
 4     for $l = 1$ to $|V^0| - 1$ do
 5         if capacity of $[S_l, V^0 \setminus S_l]$ is less than 2 then
 6             $B \leftarrow \{(i,j) : i \in S_l \cup V_u \cup V_{pu}, j \in (S_l \cup V_u \cup V_{pu}) \setminus \{i\}\}$
 7             forall $(i,j) \in B$ do $z_{ij} = 2(t_i^* + y_{ij}^*)$
 8             $(u,v) \leftarrow argmax_{(i,j) \in B}\{z_{ij}\}$
 9             if $z_{uv} > x^*(\delta(S_l))$ then
10                 $S \cup \{u,v\}$ and $u, v$ form a violated double-cut inequality
11                 $C \leftarrow C \setminus \{u\}$
12     forall $i \in C$ do
13         forall $j \in V_h \cup V_{ph} \setminus \{i\}$ do
14             Find minimum cut $[S, (V^0 \cup \{i,j\}) \setminus S]$ between $i$ and 0 on $G_{ij}^*$
15             if capacity of $[S, (V^0 \cup \{i,j\}) \setminus S]$ is less than $2(t_i^* + y_{ij}^*)$ then
                   $S \cup \{i,j\}$ and $i, j$ form a violated small-cut inequality
16 end
```

5.4.4 Extended F-partition inequalities

It was shown in Chapter 4 that the separation problem associated with the extended F-partition inequalities is NP-Hard. Therefore we have employed two heuristics to identify violated extended F-partition inequalities. We utilize the same approach for the 2ECSDHP as the structure of the extended F-partition inequalities are the same.

We start the separation algorithm by searching odd fractional cycles on the backbone of the support graph. The difference here is that we only try to separate extended F-partition inequalities if the support graph is connected. Assume that there exists such a cycle and let $\{v_1, \ldots, v_p\}$ be the set of nodes inducing a fractional odd cycle. Let $V_0 = V \setminus \{v_1, \ldots, v_p\}$. We choose edges from $\delta(V_0)$ with values greater than $\frac{1}{2}$ and put them into F in such a way that $|F|$ is odd. We check the corresponding inequality for violation and if it is violated we add it to our subproblem.

If no violated inequality is found in the first stage we start our second heuristic, which is based on finding the maximum cut in a graph. We just set the capacity of each edge e to $1 - x_e^*$ and solve the minimum cut problem as we did in the separation stage of 2ECSSP. This is because of the NP-Hardness of the maximum-cut problem. Using the algorithm of Hao and Orlin [22] on the support graph G^0 we find $|V^0|-1$ minimum cuts. Let S be a cutset obtained by this algorithm such that $0 \in S$ and $V_0 = S \cup V_u \cup V_{pu}$. Let $V \setminus S = \{v_1, \ldots, v_p\}$. Then our partition is $(V_0, v_1, \ldots, v_p)$. We construct F in the same way we did in the first heuristic in which we look for cycles. The extended F-partition inequality defined by this partition and F is checked to see if it is violated.

5.5 Variable Fixing

In this section, we propose some rules to fix some of the variables before or during the branch-and-cut algorithm. The aim of these rules is to improve the

Algorithm 7: Extended F-Partition Inequality Separation

```
   Input: $(x^*, y^*), G^0 = (V^0, E^0)$
1  begin
2      repeat
3          Find a fractional odd cycle $v_1, \dots, v_p$ such that $v_i \in V^0 \setminus \{0\}$ for
           $i = 1, \dots, p$
4          $V_0 \leftarrow V \setminus \{v_1, \dots, v_p\}$
5          Construct $F \subseteq \{e \in \delta(V_0) : x^*_e > 0.5\}$ so that $|F|$ is odd
6          Compute the violation for $F$ and the partition $(V_0, v_1, \dots, v_p)$
7      until no fractional odd cycle is found ;
8      if no violated extended $F$-partition inequality is found above then
9          Use algorithm of Hao and Orlin on $G^0$
10         foreach cut $[S, V^0 \setminus S]$ such that $0 \in S$ found in the algorithm do
11             $V_0 \leftarrow S \cup V_u \cup V_{pu}$
12             Construct $F \subseteq \{e \in \delta(V_0) : x^*_e > 0.5\}$ so that $|F|$ is odd
13             Compute the violation for $F$ and the partition $(V_0, v_1, \dots, v_p)$
               where $V \setminus V_0 = \{v_1, \dots, v_p\}$
14 end
```

performance of the solution algorithm by reducing the size of the problem to be solved. The fixing rules we suggest can be grouped into two classes. The rules in the first class only cuts some fractional solutions and do not affect the feasible solution set. On the other hand, the rules in the second group, not only remove some fractional points but also eliminate some integer feasible solutions. Here, one should note that elimination of some feasible solutions could be acceptable provided that they are not optimal, as we are trying to find the optimal solution. In addition, even if we do not use the second class of rules, the commercial LP solvers apply some preprocessing throughout the branch-and-bound algorithm, which might result in the elimination of some feasible solutions. Therefore, the second class of rules are valid if they do not omit all optimal solutions.

Unlike the number of constraints, the number of variables in the proposed mathematical model is polynomial. However, it still results in a large model when the number of nodes of the instance increases. Having a large model causes two problems. The first and obvious one is the longer Cpu times required for the solution of the linear programs solved throughout the algorithm. The second one is not as obvious as the first one and can be observed during experimentations. When the size of a linear program, the number of constraints, variables, and the non-zero terms in the constraint matrix, is large, the time we spend to add the violated inequalities found by solving the separation algorithms becomes significant. It should be noted that, by the time spent on adding the violated inequalities, we mean the length of the interval that begins with the providing of the inequalities to be added and ends by the start of the solution of the next linear problem. In other words, the solution times of the separation problems are not included.

The reason why addition of new inequalities takes significant amount time can be explained as follows. Once you provide the LP solver, which is CPLEX in our experimentations, with a linear or integer program, it first converts it into another model using its internal data structures. Naturally, it also converts the inequalities we provide into its internal representation. How this conversion performed and why it takes too long are not known as these operations are not revealed to the end-users. However, during our computational experimentations we observed that there are at least two factors that significantly affect the length of this problem. The first is the number of inequalities added at an iteration. The higher the number of inequalities you add is, the longer the addition takes. There is a trade-off here. If you add many inequalities at a time, the lower bound improves faster and less number of iterations may be sufficient but you spent more time on the addition. On the other hand, if you add fewer inequalities then you spend less time to add the inequalities but you need to make more iterations. It is observed that the performance of both strategies change for every instance, i.e., it cannot be possible to expect a better or worse performance before solving the problem.

The second factor is the preprocessing of the LP solver. As we mentioned before, commercial LP solvers have preprocessing tools which run before and/or during branch-and-bound process unless it is prevented by the user. The preprocessing tool may improve some bounds and coefficients of the variables. The elimination of some variables and/or constraints is also possible by this process.

So normally one can expect to have a tighter and smaller model after preprocessing which would possibly yield a better solution time. However, there is also a trade-off here as described before in the explanation of the effect of the number of inequalities added at each iteration. When preprocessing is applied, the model provided to the LP solver will change more and this will increase the time spent on querying the solution of a subproblem as well as the addition of new inequalities. Note that, the change after a preprocessing does not consist of the change of the data structures used. It also includes the change of the model itself. The solver may solve a different model than you provide. Although the changed model will give the same result and is convertible to the original model (in some cases the original model may not be obtained, but linear programs are not in this category) a conversion between the original and the preprocessed models is necessary. Clearly, the time lost during these operations is reasonable provided that the preprocessing improves the solution time. It may not be very easy to determine the amount of time gained by and spent for the preprocessing. However, in our case this was an easy task because of the structure of the problems. First we should note that preprocessing does not affect our initial problem, i.e., no change is done to the model, which makes it clear that preprocessing does not have any positive effect on the solution. This may be due to the fact we provide only a small part of the entire model at the beginning. In addition, all constraints are facet defining which makes it hard to improve the model by some simple preprocessing tool. For this reason, preprocessing is turned off in the initial experimentations and it is observed that performance of the branch-and-cut algorithm may increase up to 10 % in some instances, which shows that the transformations performed during preprocessing might have significant effects on the performance of the algorithm. These explanations may seem redundant as we do not use the preprocessing, however, we will revisit the importance of the preprocessing after we present the variable fixing rules.

Now consider the variable fixing rules we propose below:

1. Let $(i, j) \in A$. If $d_{ij} \geq d_{ik}$ for every $(i, k) \in A$ then we can fix $y_{ij} = 0$.

2. Let $\overline{z}$ be the objective function value of a feasible solution and $\underline{z}$ be the objective function value of the current linear program. Let $\overline{x}$ and $\overline{u}$ denote the solution and reduced cost vectors, respectively. We use $\overline{x}_i$ ($\overline{u}_i$) to represent the i^{th} component of $\overline{x}$ ($\overline{u}$).
   - If $\overline{x}_i = 0$, variable x_i is a nonbasic variable at its lower bound and if $\underline{z} + \overline{u}_i > \overline{z}$ then we can fix $x_i = 0$.
   - If $\overline{x}_i = 1$, variable x_i is a nonbasic variable at its upper bound and if $\underline{z} - \overline{u}_i > \overline{z}$ then we can fix $x_i = 1$.

3. Let $H = \{i \in V : t_i = 1\}$ be the set of nodes which are fixed to be hub nodes. If $|H| \geq 2$ then at least two of the nodes that will be a hub in the optimal solution are known. Let $i \in V \setminus H$ and $u, v \in H$ such that $d_{iu} \leq d_{ij}$ for every $j \in H$, and $d_{iv} \leq d_{ij}$ for every $j \in H \setminus \{u\}$. Then for every $j \in V \setminus \{i\}$ such that $d_{ij} > d_{iv}$ we can fix $y_{ij} = 0$.

4. Let $\{i, j\} \in E$. If x_{ij} is fixed to 1, then we can fix $y_{ik} = 0$ for every $k \in V \setminus \{i\}$, $y_{jk} = 0$ for every $k \in V \setminus \{j\}$, and $t_i = t_j = 1$.

5. Let $(i, j) \in A$. If y_{ij} is fixed to 1, then we can fix $y_{jk} = 0$ for every $k \in V \setminus \{j\}$, $y_{ki} = 0$ for every $k \in V \setminus \{i, 0\}$, $x_e = 0$ for every $e \in \delta(i)$, $t_i = 0$, and $t_j = 1$.

6. Let $i \in V \setminus \{0\}$. If t_i is fixed to 1, then we can fix $y_{ij} = 0$ for every $j \in V \setminus \{i\}$.

7. Let $i \in V \setminus \{0\}$. If t_i is fixed to 0, then we can fix $y_{ji} = 0$ for every $j \in V \setminus \{i, 0\}$ and $x_e = 0$ for every $e \in \delta(i)$.

First fixing rule is used once at the beginning of the algorithm. As there will be at least three hubs in a feasible solution of the problem, a user will not be assigned to a hub with the highest assignment cost. Second rule is a well known one for variable fixing and uses the reduced cost information [44]. A feasible solution is necessary to apply this rule and clearly better feasible solutions will possibly allow more fixing. Rule 3 depends on the fact that local access network design, i.e., problem of assigning the users to hubs is trivial when the hubs are known. Users are assigned to hubs with the least assignment costs. According to rule 3, if we know the locations of at least two hubs from the information obtained via variable fixing, then we can choose two which offer the least cost and then we can conclude that a node will not be assigned to another hub with a higher assignment cost. First three rules are based on optimality conditions, however, the remaining ones, Rules 4-7, are based on the relation inequalities. Actually they are implied by the formulation. However, not including all relation inequalities in the subproblems prevent us to use these implications. Therefore, using them we can fix some variables before corresponding constraints are added to the model.

Rule 2 is used at each step after solving a subproblem and we keep applying Rules 3-7 until we cannot fix a new variable. Once a variable is fixed it can be removed from the model.

Variable fixing provides several advantages. The first is the reduction of the size of the model. The second is that we can identify some constraints that become redundant after fixing a particular variable. Let $S \subset V \setminus \{0\}$ be a node set and $i \in S$. Clearly, S and i define a cut inequality. Note that for every $j \in S \setminus \{i\}$ there is another cut inequality defined by S and j. So there are $|S|$ cut inequalities which are induced by the same node set and a feasible solution must satisfy all of them. Now suppose that the variable t_i is fixed to 1 during the solution algorithm. Clearly, the cut inequalities with S are not required anymore as $x(\delta(S)) \geq 2$ becomes valid. So all the cut inequalities can be removed that were added until this point. This reduces the size of the model and so the second advantage is indirectly related to the first one. Finally, if sufficiently many variables can be fixed, then the preprocessor of the LP solver can identify some possible improvements and makes additional modifications. The effects of the variable fixing are discussed in the next Section where we present our computational experiments.

5.6 Computational results

In order to find the optimal solution we developed a branch-and-cut algorithm based on the separation algorithms described in the previous section. Before providing the details of the branch-and-cut implementation and the computational results, we want to discuss some modeling alternatives and their strengths.

Consider the proposed formulation. In this model 2-edge connectivity of the backbone network is ensured by the cut inequalities (5.29). Note that the double-cut (5.37) and small-cut (5.36) inequalities can also be used instead of the cut inequalities. In the previous section we have shown that all three inequality classes define facets. In order to compare the LP relaxation values i.e., the strengths of the formulations a branch-and-cut algorithm is developed for each three formulation. These algorithms are implemented in C++ using the Concert technology 29 to manage the branch-and-cut tree. As these results are just used to get initial information about the formulations, the algorithms are not tuned to obtain better performance.

Before presenting the table, we explain some columns that appears in most of the tables below:

Opt optimal objective function value;

LPbound LP relaxation value at the root node of the branch-and-cut tree;

Gap the relative error between the best upper bound (the optimal solution value if the problem is solved to optimality) and the lower bound obtained at the root node of the branch-and-cut tree;

NR number of generated relation inequalities;

NCut number of generated cut inequalities;

NDb number of generated double-cut inequalities;

NDa number of generated double-assignment inequalities;

NFP number of generated extended F-partition inequalities;

NNode number of branch-and-cut nodes evaluated;

Cpu total CPU time in seconds (rounded to the nearest integer);

In addition, we provide the name of the instance that includes the number of nodes in the graph and the α value in the first two columns of the tables.

In Tables 5.1 and 5.2, we provide the LP relaxation values of the alternative formulations. The columns db bnd, sm bnd and cut bnd are the LP relaxation values obtained from the formulations with the double-cut, small-cut, and cut inequalities, respectively. To make the comparisons easier we also provide these values in another form. We divide each LP bound with the greatest of three LP bounds and multiply with 100. By this way, the greatest LP bound becomes 100 and the other ones take values accordingly. This allows easier interpretation of the values. It can be seen that the formulation with the cut inequalities provides the strongest formulation in overall. For $\alpha = 3$ all formulations provide the same bound. This is reasonable as in instances with $\alpha = 3$ all nodes become hubs, so the cut based inequalities provide the same lower bounds on the number of edges in the cutsets. As α increases, the formulation with the double-cuts gets weaker quickly. Especially when $\alpha = 9$ this formulation provides very weak lower bounds. This can be explained with the double-cut inequality defined by a set with only one node. Let $S = \{i\}$ and i and $j \in V \setminus S$ be the fixed nodes. Consider the corresponding cut inequality.

$$x(\delta(i)) \geq 4t_i + 2y_{ij} - 2 \tag{5.40}$$

It can be seen that the right hand side can be arranged in such a way that it is not positive. This means in practice node i is a hub with positive $t_i > 0$ but there is no edge attached to it so i is not on the backbone network. This results in low t values for many nodes and hence a very weak lower bound is obtained. The formulation with the small-cut inequalities provides almost the same lower bound as the cut inequality formulation does for $\alpha \in \{3, 5\}$. Starting with $\alpha = 7$

it also gets weaker. It can be observed that for some instances with $\alpha = 5$ it provides the best bound. However, the difference is very small in these instances and it is likely to be due to numerical inaccuracies.

So the cut inequality formulation is the best one and hence we decided to use it in our computational analysis and the other inequalities are used as valid inequalities to improve the LP relaxation values at the root node of the branch and cut tree.

Deciding the formulation we will use, we continued our experimentation. Based on our polyhedral analysis and the separation algorithms described earlier we developed a branch-and-cut algorithm for the dual homing problem. We start the optimization by solving the following linear program:

$$
\begin{aligned}
z = \sum_{i \in V} d_{ii} + \min \sum_{e \in E} c_e x_e + \sum_{(i,j) \in A} d'_{ij} y_{ij} & & \\
\text{s.t. } 2x_{0i} + \sum_{k \in V \setminus \{i\}} y_{ik} \le 2 & \qquad & \forall i \in V \setminus \{0\} \\
x(\delta(i)) + \sum_{j \in V \setminus \{i\}} y_{ij} \ge 2 & & \forall i \in V \setminus \{0\} \\
x(\delta(0)) \ge 2 & & \\
x(\delta(\{0, i\})) \ge 2 & & \forall i \in V \setminus \{0\} \\
0 \le x_e \le 1 & & \forall e \in E \\
0 \le y_{ij} \le 1 & & \forall (i,j) \in A.
\end{aligned}
$$

The solution $(\overline{x}, \overline{y})$ of this initial subproblem is feasible for the dual homing problem if it satisfies the cut inequalities, the relation inequalities and the integrality constraints. Therefore at each iteration of the branch-and-cut algorithm we solve the separation problems to determine if there are violated inequalities. The separation procedures for different classes of inequalities are performed in the following order: relation (5.26), cut (5.29), and double-cut (5.37) inequalities. We generate up to 200 violated valid inequalities at each iteration. We generate the inequalities in the given order and if the number of violated relation

Instance	α	db bnd	sm bnd	cut bnd	db p	sm p	cut p
eil101	3	1882.5	1882.5	1882.5	100	100	100
eil101	5	2729.04	2996.5	2996.5	91.07	100	100
eil101	7	1602.43	3079.22	3095.67	51.76	99.47	100
eil101	9	701.78	1738.49	2060.75	34.05	84.36	100
eil51	3	1267.5	1267.5	1267.5	100	100	100
eil51	5	1907.86	2048.5	2048.5	93.13	100	100
eil51	7	1142.13	2124.34	2225.57	51.32	95.45	100
eil51	9	575.75	1278.99	1316	43.75	97.19	100
gr96	3	162082	162082	162082	100	100	100
gr96	5	236486	255605	255538	92.52	100	99.97
gr96	7	139544	280237	282923	49.32	99.05	100
gr96	9	52787.3	163983	236995	22.27	69.19	100
kroA100	3	62809.5	62809.5	62809.5	100	100	100
kroA100	5	97923.7	101310	101310	96.66	100	100
kroA100	7	59069.4	119474	119841	49.29	99.69	100
kroA100	9	25143.1	71898.6	97653.5	25.75	73.63	100
kroA150	3	78897	78897	78897	100	100	100
kroA150	5	121783	127978	127975	95.16	100	100
kroA150	7	73038.5	145587	146752	49.77	99.21	100
kroA150	9	26578.8	84175.6	117170	22.68	71.84	100
kroB100	3	65502	65502	65502	100	100	100
kroB100	5	102250	106259	106259	96.23	100	100
kroB100	7	55395.1	122760	123747	44.76	99.2	100
kroB100	9	21331.1	66240.6	97171.1	21.95	68.17	100
kroB150	3	77186.5	77197.5	77197.5	99.99	100	100
kroB150	5	118732	124562	124523	95.32	100	99.97
kroB150	7	61554.8	140308	140996	43.66	99.51	100
kroB150	9	23748.6	79442.9	113105	21	70.24	100
kroC100	3	61417.5	61417.5	61417.5	100	100	100
kroC100	5	97731.2	101266	101266	96.51	100	100
kroC100	7	50234.7	117299	118057	42.55	99.36	100
kroC100	9	19937.6	68059.8	95066	20.97	71.59	100
kroD100	3	63424.5	63424.5	63424.5	100	100	100
kroD100	5	99453.4	103312	103296	96.27	100	99.98
kroD100	7	54167	109936	120948	44.79	90.9	100
kroD100	9	23252.3	67695.6	94899	24.5	71.33	100
kroE100	3	65398.5	65398.5	65398.5	100	100	100
kroE100	5	101886	106267	106220	95.88	100	99.96
kroE100	7	52280.9	121702	122457	42.69	99.38	100
kroE100	9	18996.6	69814.5	98769.8	19.23	70.68	100

Table 5.1: LP bounds of alternative formulations.

Instance	α	db bnd	sm bnd	cut bnd	db p	sm p	cut p
lin105	3	43111.5	43111.5	43111.5	100	100	100
lin105	5	68006.9	70342.5	70342.5	96.68	100	100
lin105	7	34731.9	85414.2	87064	39.89	98.11	100
lin105	9	13690.4	49979.1	71083	19.26	70.31	100
rat99	3	3618	3618	3618	100	100	100
rat99	5	5734.59	5955.25	5952.75	96.29	100	99.96
rat99	7	3134.43	6816.23	6846.65	45.78	99.56	100
rat99	9	1372.77	4198.51	5386.38	25.49	77.95	100

Table 5.2: LP bounds of alternative formulations cont'd.

inequalities is less than 200 we also generate violated cut inequalities. But we generate double cut inequalities only if we cannot find any violated cut or relation inequality. Note that if a solution is integral and there is no violated cut or relation inequality then the solution is feasible and there is no need to solve the separation problem for the double-cut inequalities.

Our test problems are based on TSP instances from TSPLIB 2.1 [38]. To compute the backbone link setup costs and assignment costs we use the following formulas, $c_{ij} = \lceil \alpha l_{ij} \rceil$ and $d_{ij} = \frac{\lceil (10-\alpha) l_{ij} \rceil}{2}$ where l_{ij} denotes the distance between nodes i and j in the TSPLIB instances and $\alpha \in \{3, 5, 7, 9\}$. We set $d_{ii} = 0$ for all $i \in V \setminus \{0\}$. Note that as α decreases, assignment costs increase while backbone link setup costs decrease, i.e., the problem gets closer to the 2-edge connected subgraph problem. Conversely, as α increases the number of nodes chosen to be hubs decreases and the problem gets farther from the 2-edge connected subgraph problem.

A construction heuristic is used to find an initial solution at the beginning of the algorithm. We start from node 0 and apply the nearest neighbor TSP heuristic to obtain a cycle that includes all nodes of the graph. As this cycle is 2-edge connected, it forms a feasible solution. Throughout the branch-and-cut algorithm, we also use an LP based improvement heuristic. The edges are ranked in a decreasing order according to their values in the fractional solution. We start with an empty set and add the edges one by one until we obtain a 2-edge connected graph.

We implemented our algorithm in C++ using Concert Technology 29 as the framework and CPLEX 12.1 as the LP solver. Computational analysis is performed on a workstation with 2.66 GHz xeon processor and 8 Gb of memory. We use the default strategies of CPLEX in searching the branch-and-cut tree. Tailing off control is used; we branch if the improvement in the objective function value is small in 10 subsequent iterations.

Some computational results are provided in Table 5.3. In this table, Rate columns denotes the percentage of the nodes selected to be hubs in the optimal solution.

We do not go into the details of these results because we have even improved the branch-and-cut algorithm by incorporating a variable fixing scheme. So we first want to give the results obtained by the new method. These computational results are given in Table 5.4. In the last two columns we give the ratio of the number of variables we fixed before branching starts to the total number of variables and the improvement in the CPU time obtained by the new method in percentage, respectively. From the FixRate column it can be seen that the variable fixing scheme we develop makes it possible to fix many variables before starting branching and this improves the performance of the branch-and-cut. The improvement obtained by variable fixing is significant as it can be seen from the CpuImp columns. Therefore we make the main computational analysis using the branch-and-cut algorithm with variable fixing. The details of this algoritm can be found in Algorithm 8.

We solved 20 instances which has 150 - 198 nodes. We first used only relation and cut inequalities in the branch-and-cut algorithm. All instances are solved to optimality in less than 9000 seconds. The largest gap between the optimal solution values of the instances and the LP relaxation values is 1.53%. The details of the results are presented in Table 5.5.

To observe the effects of the double-assignment inequalities to the branch-and-cut algorithm we solved the same 20 instances using the relation, cut and

Instance	α	Rate	Opt	LPbound	Gap	NR	NCut	Cpu	NNode
eil101	3	100	1887	1882.5	0.24	10	389	6	5
eil101	5	83	2997	2996.5	0.02	156	1167	22	2
eil101	7	42	3118	3095.67	0.72	597	2865	110	103
eil101	9	19	2062	2060.75	0.06	1755	1857	79	3
gr96	3	100	163935	162082	1.13	51	1247	14	128
gr96	5	91	257729	255538	0.85	177	2475	27	82
gr96	7	55	284152	282923	0.43	455	1282	27	13
gr96	9	33	237636	236995	0.27	1507	2106	159	11
kroA100	3	100	63783	62809.5	1.53	45	869	16	402
kroA100	5	88	102203	101310	0.87	164	429	6	79
kroA100	7	64	120159	119841	0.26	445	748	12	18
kroA100	9	24	97663	97653.5	0.01	1525	1800	61	2
kroB100	3	100	66177	65502	1.02	32	697	11	63
kroB100	5	90	107189	106259	0.87	173	3613	61	243
kroB100	7	54	123863	123747	0.09	466	1308	25	8
kroB100	9	20	97443	97171.1	0.28	1657	2003	232	19
kroC100	3	100	62247	61417.5	1.33	52	1162	15	92
kroC100	5	91	102378	101266	1.09	166	1878	26	76
kroC100	7	56	118387	118057	0.28	467	1334	21	16
kroC100	9	24	95066	95066	0	1466	1671	50	0
kroD100	3	100	63882	63424.5	0.72	38	926	15	18
kroD100	5	88	103387	103296	0.09	161	968	13	4
kroD100	7	58	121198	120948	0.21	469	1312	26	15
kroD100	9	24	95142	94899	0.26	1514	2194	142	11
kroE100	3	100	65769	65398.5	0.56	48	791	5	39
kroE100	5	91	107251	106220	0.96	174	5867	73	76
kroE100	7	56	122555	122457	0.08	466	848	16	3
kroE100	9	29	99685	98769.8	0.92	1552	2897	262	41
lin105	3	100	43137	43111.5	0.06	56	777	11	2
lin105	5	91	70403	70342.5	0.09	160	471	5	3
lin105	7	69	87064	87064	0	447	1743	25	0
lin105	9	36	71122	71083	0.05	1666	2334	109	2
rat99	3	100	3633	3618	0.41	12	536	8	13
rat99	5	93	5965	5952.75	0.21	169	1525	23	27
rat99	7	45	6854	6846.65	0.11	539	1894	38	11
rat99	9	22	5397	5386.38	0.2	1702	2646	206	10

Table 5.3: Initial computations.

Instance	α	LPbound	Gap	NR	NCut	Cpu	NNode	FixRate	CpuImp
eil101	3	1882.5	0.24	10	411	1	6	87.6	89.1
eil101	5	2996.5	0.02	149	982	1	0	99.5	95.1
eil101	7	3095.67	0.72	580	1253	108	117	3.2	1.4
eil101	9	2060.75	0.06	1635	1978	38	1	87.1	52.1
gr96	3	162082	1.13	47	455	1	239	79.2	90.8
gr96	5	255538	0.85	198	762	5	168	50.8	81.7
gr96	7	282923	0.43	447	1244	19	19	10.7	28.5
gr96	9	236995	0.27	1436	2453	94	14	68.1	40.9
kroA100	3	62809.5	1.53	42	329	2	450	71.6	86.3
kroA100	5	101310	0.87	176	497	2	104	76.5	72.9
kroA100	7	119841	0.26	445	1159	10	14	19.2	13.6
kroA100	9	97653.5	0.01	1435	2145	43	0	93.9	28.7
kroB100	3	65502	1.02	33	616	2	73	62.5	81.5
kroB100	5	106259	0.87	181	1899	20	406	35.4	67.6
kroB100	7	123747	0.09	455	1484	16	4	10.3	34.1
kroB100	9	97171.1	0.28	1579	2559	134	4	17.9	42.3
kroC100	3	61417.5	1.33	56	694	3	136	69.2	77.6
kroC100	5	101266	1.09	173	1357	7	208	37.7	73.5
kroC100	7	118057	0.28	455	1422	12	17	33.3	43.4
kroC100	9	95066	0	1472	2045	35	1	10.8	30.5
kroD100	3	63424.5	0.72	38	802	2	35	73.8	88.8
kroD100	5	103296	0.09	155	931	2	6	88.7	87.0
kroD100	7	120948	0.21	466	1360	24	16	3.1	6.5
kroD100	9	94899	0.26	1477	2569	139	30	24.5	2.4
kroE100	3	65398.5	0.56	48	552	1	62	89.3	87.9
kroE100	5	106220	0.96	189	2314	17	303	47.0	76.6
kroE100	7	122457	0.08	457	1233	9	3	17.2	42.4
kroE100	9	98769.8	0.92	1585	2714	245	95	25.8	6.4
lin105	3	43111.5	0.06	57	779	1	2	93.0	90.3
lin105	5	70342.5	0.09	162	743	1	6	61.7	71.5
lin105	7	87064	0	461	1953	6	1	99.8	76.4
lin105	9	71083	0.05	1538	2823	99	2	19.1	9.0
rat99	3	3618	0.41	14	546	1	33	74.5	90.7
rat99	5	5952.75	0.21	173	1212	4	47	32.6	81.8
rat99	7	6846.65	0.11	522	2002	22	8	22.4	43.2
rat99	9	5386.38	0.2	1651	3394	168	39	13.9	18.5

Table 5.4: Initial computations with variable fixing.

Algorithm 8: Branch-and-cut algorithm with variable fixing

```
Input: LP : initial linear program, nbvariables : number of variables in LP
1  begin
2      Step 0: Apply initial variable fixing
3      Step 1: Solve LP
4      Step 2: Apply variable fixing
5      Step 3: if number of fixed variables $\geq \min\{0.1 nbvariables, 3000\}$ then
6          Reconstruct LP without fixed variables
7          Provide the basis information from the last solution
8          Go to Step 1
9      else
10         Add variable fixing constraints to LP
11     Step 4: Apply separation algorithms
12     Step 5: if violated inequalities are found then
13         Add inequalities to LP
14         Go to Step 1
15     else if All variables are integral then
16         Optimal solution is found. Stop
17     Step 6: Apply variable fixing
18     Step 7: Reconstruct LP without fixed variables and solve using the
       basis information from the last solution
19     Step 8:  while There are branch-and-cut nodes to be exploited do
20         repeat
21             Apply separation algorithms
22             if violated inequalities are found then
23                 Add violated inequalities to LP
24                 Solve LP
25         until No violated inequality is found ;
26         if All variables are integral then  Prune node else  Branch on a
           fractional variable
27 end
```

Instance	α	LPbound	Gap	NR	NCut	Cpu	NNode	FixRate
kroA150	3	78897	0.85	59	1701	10	494	79.2
kroA150	5	127975	0.42	262	2004	19	85	46.4
kroA150	7	146752	0.3	694	1979	72	18	13.9
kroA150	9	117170	0.07	2193	3693	541	7	22.4
kroB150	3	77197.5	1.26	90	2362	37	1061	53.4
kroB150	5	124523	1.50	322	8213	1861.5	17888	30.0
kroB150	7	140995	0.13	742	3331	36	4	48.3
kroB150	9	113105	0.19	2347	4353	712	18	17.5
pr152	3	219626	0.64	82	2132	12	104	84.8
pr152	5	363614	0.71	298	3995	144	1084	40.1
pr152	7	466432	1.07	793	4132	2778	22550	9.5
pr152	9	484689	0.47	2330	6651	6774	924	6.3
u159	3	125775	0.37	50	1195	5	51	79.9
u159	5	206486	0.34	259	8130	37	62	70.9
u159	7	243440	0.22	770	3348	92	19	16.3
u159	9	203646	0.35	2493	5622	1553	184	20.8
d198	3	47136	0.43	44	4835	154	554	47.5
d198	5	77696.5	0.1	343	10887	212	20	37.2
d198	7	96606.4	0.22	1080	9589	1311	503	18.9
d198	9	97397.7	0.06	3320	15857	8522	54	2.7

Table 5.5: Larger Instances with relation and cut inequalities.

Instance	α	NR	NCut	NDa	Cpu	NNode	fix	bnd	cpu	nd
kroA150	3	56	1726	36	10	489	79,3	0	-3,4	1,0
kroA150	5	254	1494	152	9	64	74,2	0,6	53,2	24,7
kroA150	7	719	2260	316	49	2	15,2	46,4	31,9	88,9
kroA150	9	2252	3460	215	495	2	23,9	60	8,4	71,4
kroB150	3	86	2063	52	37	1002	53,4	0	-0,8	5,6
kroB150	5	329	15107	208	2822	17045	15,4	9.0	-51.6	4.7
kroB150	7	739	2721	295	22	1	99,8	100	38,5	75,0
kroB150	9	2330	4533	184	910	12	17,4	23,8	-27,7	33,3
pr152	3	82	2132	46	12	116	84,8	0	-1,2	-11,5
pr152	5	273	4847	168	71	242	57,9	14,7	51	77,7
pr152	7	802	5020	528	1923	4431	9,5	12,8	30,8	80,4
pr152	9	2225	6380	451	4593	478	6,2	14	32,2	48,3
u159	3	46	1376	23	8	46	80	0	-49,4	9,8
u159	5	251	8946	163	56	37	56,1	8,4	-51,3	40,3
u159	7	765	3155	310	59	9	15,9	54,8	35,8	52,6
u159	9	2548	6124	231	1787	161	21,6	3,6	-15,1	12,5
d198	3	46	4631	25	118	470	55,3	0	23,4	15,2
d198	5	339	49682	199	1758	20	32,8	9,6	-729,9	0,0
d198	7	1101	9722	508	1434	205	7,8	53,9	-9,4	59,2
d198	9	3464	18540	274	5629	15	33,1	62,3	33,9	72,2

Table 5.6: Larger Instances with relation, cut and double assignment inequalities.

double-assignment inequalities. We provide the results in Table 5.6. The last three columns of Table 5.6, namely bnd, cpu, and nd, show the percent differences in the LP relaxation gaps, solution times and number of branch-and-cut nodes exploited, respectively. In these columns, positive values indicate improvements and negative values are used if the measure got worse. It is observed that the double-assignment inequalities are not useful when $\alpha = 3$. Although we generated some double-assignment constraints they do not affect the LP relaxation values. In other cases the LP bounds are improved significantly and one instance is solved to optimality in the root node. The number of exploited branch-and-cut nodes also decreased in general. However, there is also one instance in which more nodes are analyzed by the LP solver. In 10 instances the solution times improved, while in 8 instances the Cpu times increased significantly. In 2 instances the effects can be ignored.

We tried to improve the performance of the branch-and-cut algorithm by including the double-cut inequalities in the solution process of our problem instances. In this experiment we still have the relation, cut and double-assignment inequalities. It is observed that in 6 instances the Cpu times improved significantly, while in 3 instances we obtained worse results. In the remaining 9 instances the differences are not significant. We also observed that the double-cut inequalities are useful for improving the LP relaxation values especially when $\alpha = 9$. 3 instances are solved to optimality without branching by using the double-cut inequalities. However, no violated double-cut inequality could be found when $\alpha \in \{3, 5\}$ and hence they do not have any effect in these cases. The number of branch-and-cut nodes exploited also decreased in general, however, in two instances the number increased. The results can be found in Table 5.7 and the last three columns are used to denote the percent differences in the LP relaxation gaps, solution times and number of branch-and-cut nodes exploited, respectively, as in the previous table.

Finally, we include the extended F-partition inequalities in the computational experiments. We present the results of this experiment in Table 5.8. It is observed that the extended F-partition inequalities significantly improve the LP relaxation bounds in most of the instances. Besides, in two instances, it seems that there is no effect of these inequalities, however, these instances are already solved to optimality without any branching. The effect of the extended F-partition inequalities on solution times is similar. In 13 instances we obtain improvement in Cpu times while only in 2 instances the solution times increased. Although the number of branch-and-cut nodes also decreased in most instances there are significant increases in some instances.

In addition, comparing these results to the ones obtained when only the relation and cut inequalities, it can be seen that in all instances except one, the solution times reduced, which shows that the addition of valid inequalities improve the performance of the branch-and-cut algorithm.

In the computational experiments we also included the small-cut inequalities. Although, the small-cut inequalities provide strong LP relaxation bounds like

Instance	α	NR	NCut	NDa	NDb	Cpu	Node	bnd	cpu	nd
kroA150	3	56	1726	36	0	10	489	0	0,7	0
kroA150	5	254	1494	152	0	9	64	0	0,9	0
kroA150	7	723	2260	326	3	54	8	8	-10,8	-300
kroA150	9	2252	3460	214	3	408	1	100	17,5	50
kroB150	3	86	2063	52	0	38	1002	0	-0,3	0
kroB150	5	329	15107	208	0	2789	17045	0	1,1	0
kroB150	7	739	2721	295	0	22	1	0	1,5	0
kroB150	9	2323	4613	181	33	458	1	100	49,7	92
pr152	3	82	2132	46	0	13	116	0	-1,2	0
pr152	5	273	4847	168	0	71	242	0	0,1	0
pr152	7	794	5214	528	10	2863	7061	3,9	-48,9	-59
pr152	9	2154	10262	380	2684	6429	7	72,6	-40	98
u159	3	46	1376	23	0	8	46	0	0,9	0
u159	5	251	8946	163	0	55	37	0	0,4	0
u159	7	763	3156	310	25	45	2	57,5	23,1	78
u159	9	2533	6826	226	218	1413	35	68	20,9	78
d198	3	46	4631	25	0	120	470	0	-1,9	0
d198	5	339	49682	199	0	1759	20	0	0	0
d198	7	1087	10404	487	78	785	59	10,6	45,3	71
d198	9	3298	15311	229	81	3510	1	100	37,6	93

Table 5.7: Larger Instances with relation, cut, double assignment and double-cut inequalities.

Instance	α	NCut	NDa	NDb	NFP	Cpu	Node	bnd	cpu	nd
kroA150	3	961	30	0	73	8	74	40.3	20.0	84.9
kroA150	5	2426	161	0	186	8	2	99.5	11.1	96.9
kroA150	7	3468	434	16	97	54	2	47.9	0.0	75.0
kroA150	9	3460	214	3	0	408	1	0.0	0.0	0.0
kroB150	3	1364	36	0	69	8	95	34.7	78.9	90.5
kroB150	5	9837	213	0	98	232	797	39.0	91.7	95.3
kroB150	7	2207	286	0	29	15	1	0.0	31.8	0.0
kroB150	9	5144	178	33	13	573	1	0.0	-25.1	0.0
pr152	3	2300	46	0	39	11	164	7.0	15.4	-41.4
pr152	5	3094	213	0	50	46	265	14.8	35.2	-9.5
pr152	7	6611	653	10	105	1137	1411	10.2	60.3	80.0
pr152	9	9361	274	2317	79	6219	4	0.7	3.3	42.9
u159	3	720	22	0	41	4	3	45.4	50.0	93.5
u159	5	5325	161	0	117	14	1	100.0	74.5	97.3
u159	7	3067	367	43	39	51	3	89.8	-13.3	-50.0
u159	9	6454	183	218	28	1219	15	0.8	13.7	57.1
d198	3	4093	21	0	83	106	63	36.0	11.7	86.6
d198	5	38770	202	0	259	706	1	100.0	59.9	95.0
d198	7	10269	651	63	16	829	88	10.5	-5.6	-49.2
d198	9	15449	218	81	24	3622	1	0.0	-3.2	0.0

Table 5.8: Larger Instances with relation, cut, double assignment, double-cut and extended F-partition inequalities.

the cut inequalities, we observed that these inequalities are not effective in the solution of 2ECSDHP. There are two observations on the small-cut inequalities. Note that in the experiments we first separate the relation and cut inequalities which are the constraints of our formulation. The valid inequalities are separated if sufficient number of violated constraints could not be found. If the small-cut inequalities are separated after all cut inequalities are satisfied, then no violated small-cut inequalities could be found. If we give higher priority to the small-cut inequality separation, i.e., we separate them before cut inequalities, then many violated small-cut inequalities can be found. However, in this case Cpu times get significantly worse. Therefore we do not include the small-cut inequalities in this analysis.

It can be seen that the instances used to solve 2ECSSP and 2ECSDHP are not exactly the same. The reason of this difference is that, our initial experimentations showed that 2ECSDHP is more difficult than 2ECSSP to solve. Therefore, we could not increase the size of the problems as we did for 2ECSSP. Therefore, in order to increase the number of instances solved, we added new instances with less than 200 nodes to our experiments.

5.7 Conclusion

We analyzed a variant of the 2ECSSP in which the survivability is extended to the local access networks in this chapter. Different formulations were developed and compared. A polyhedral analysis was performed for the polyhedra associated with the formulation we decided to use in the solution method. Although the formulations for the 2ECSSP and 2ECSDHP are quite similar, we observed that there are significant differences in the polyhedral structures. We provided some valid inequalities, some of which are obtained by modifying the valid inequalities for 2ECSSP, while some were obtained via projection method. We observed that the solutions of the LP relaxations have more fractional values than the solutions of 2ECSSP. Since the reduction operations are more effective when the solution has less number of fractional values, we did not extend the reduction operations to the 2ECSDHP. Instead we focused on the variable fixing rules to improve the performance of the branch-and-cut algorithm.

Problem instances up to 200 nodes were solved in the computational analysis. From these results, we observed that the valid inequalities and variable fixing rules have improving effects on the performance of the proposed algorithm.

Chapter 6

Regenerator Placement Problem

In the previous chapters, we discussed the importance of the survivability and analyzed two variants of a two level survivable network design problems. In the design problems, the objective is to minimize the setup cost of the network. However, once the network is designed, other factors such as routing become more important. Since the backbone networks usually cover a large area, the distances between the concentrators of the network could be high and this may bring the necessity of signal regeneration. In such cases the routing problem is not a trivial one since some paths on the network may not be usable due to regeneration constraints. The regenerator placement problem we analyze in this chapter deals with the signal degradation which occurs especially in optical networks. In fact, signal degradation also exists in all networks, however the regeneration is more complicated in optical networks, so that the minimization of the number of regenerators used on a network becomes more important. The differences between the regeneration in electrical and optical networks were described in Chapter 2. In RPP we focus on the routing and location problems. It should be noted that, although the underlying graph is 2-edge connected, due to regeneration constraints some paths may not be used. Therefore, we also take the survivability into account by making sure that at least two edge disjoint paths are available between every node pair while solving the location problem.

In this problem we consider a weighted undirected graph $G = (N, E)$, where N

is the node set, E is the edge set of G. There is a weight c_{ij} associated with every edge $e \in E$. Note that as the edges are undirected we assume $c_{ij} = c_{ji}$ for every $e = \{i, j\} \in E$. This weight represents the amount of degradation through the edge and can be interpreted as the length of the edge as the degradation amount depends on the length the signal travels. As many node pairs on the graph communicate with each other, we use a set of commodities $K = \{1, \ldots, |K|\}$ to denote the node pairs. For every $k \in K$, there is an associated pair of nodes (s_k, t_k) representing the origin and destination nodes of commodity k, respectively. A directed path (cycle) is denoted by P (C) and $|P|$ $(|C|)$ shows the number of arcs in P (C). $L(P)$ represents the length of P, i.e., $L(P) = \sum_{(i,j)\in P} c_{ij}$. We use $R_{\max}$ to denote the maximum allowable length (degradation) of a transparent path segment, i.e., $R_{\max}$ is the maximum distance beyond which the signal has to be regenerated. Since we are focusing on regenerating the signals before they are lost, the distance which the signal traveled after the last regeneration is more important than the distance traveled after initial emission. The new distance concept will be referred to as the *real length* and is denoted by $\overline{L}$. We define $\overline{L}$ as follows.

Definition 6.1 *Let $R \subseteq N$ be a given regenerator set and $k \in K$ be a given commodity. Consider a path P from s_k to t_k. The real length of this path is $\overline{L}(P) = \max\{l_1, \ldots, l_m\}$ where l_i for $i \in [1, \ldots, m]$ for $m \geq 1$ is the length of the i^{th} segment of P between two consecutive nodes from $R \cup \{s_k, t_k\}$.*

We make two other definitions which we shall resort to throughout the text, before we formally define the problem.

Definition 6.2 *A given commodity $k \in K$ is called feasible with respect to a given regenerator set $R \subseteq N$ if there exist two edge disjoint paths P_1 and P_2 between s_k and t_k such that $\overline{L}(P_1) \leq R_{max}$ and $\overline{L}(P_2) \leq R_{max}$.*

To make a distinction between the two edge disjoint paths, one is called the *working path* and the other will be referred to as the *restoration path.*

Definition 6.3 *A given regenerator set $R \subseteq N$ is called feasible if every commodity $k \in K$ is feasible with respect to R.*

Given a weighted graph $G = (N, E)$ with weights c_e, a set of commodities, and a positive scalar R_{max}, the objective in RPP is to find a minimum cardinality feasible node subset $R \subseteq N$. Therefore, we need to find a node set R and two edge disjoint paths, P_k^1 and P_k^2, between every origin destination s_k and t_k (for every $k \in K$) that are feasible with respect to R.

6.1 Modeling the Problem

An integer linear program is formulated for the problem. Since directions are important in signal transmission, $G = (N, E)$ is transformed into a directed graph $D = (N, A)$ by replacing each undirected edge $\{i, j\} \in E$ by two directed arcs (i, j) and (j, i), both with cost c_{ij} (see [1]). The model is for finding the nodes where the regenerators should be placed and simultaneously determining two edge disjoint paths between the source and destination nodes of every commodity. We assume without loss of generality that $c_e \leq R_{max}$ $\forall e \in E$ where $R_{\max}$ is the degradation limit. As a regenerator can be used by more than one commodity, the problem has to be solved for all commodities simultaneously. We use the following decision variables:

$$x_{ijl}^k = \begin{cases} 1, & \text{if } l^{th} \text{ path for commodity } k \text{ includes arc } (i,j) \\ 0, & \text{otherwise} \end{cases}$$

Here $l = 1$ represents the working path while $l = 2$ is used to denote the restoration path.

$$r_i = \begin{cases} 1, & \text{if a regenerator is placed on node } i \\ 0, & \text{otherwise} \end{cases}$$

π_{il}^k : length of the transparent segment of path l leaving node i for commodity k.

For π's to take on proper values, we need to force the following relationship for every $(i, j) \in A$, $k \in K$ and $l \in \{1, 2\}$:

$$\pi^k_{jl} \geq (\pi^k_{il} + c_{ij})x^k_{ijl}(1 - r_j) \tag{6.1}$$

Indeed, if there is a regenerator at node j, then node j becomes the starting point of a transparent segment so π^k_{jl} is allowed to take value 0. Similarly, if $x^k_{ijl} = 0$ then nothing should be imposed on π^k_{jl}. If $r_j = 0$ and $x^k_{ijl} = 1$, then the length of the path leaving j must be greater or equal to the sum of the length of the path leaving node i and length of arc (i, j).

One can use standard big-M type of linearization for constraint (6.1) to derive a linear integer program for the regenerator placement problem. Let M1 be the model under investigation:

$$\text{minimize} \sum_{i \in N} r_i \tag{6.2}$$

subject to:

$$\sum_{j:(i,j)\in A} x_{ijl}^k - \sum_{j:(j,i)\in A} x_{jil}^k = \begin{cases} 1, & i = s_k; \\ -1, & i = t_k; \\ 0, & i \neq s_k, t_k; \end{cases} \qquad \forall i \in N, \forall k \in K, l = 1,2 \tag{6.3}$$

$$\pi_{jl}^k \geq \pi_{il}^k + c_{ij}x_{ijl}^k - R_{\max}(1 - x_{ijl}^k) - R_{\max}r_j, \qquad \forall (i,j) \in A, \forall k \in K, l = 1,2 \tag{6.4}$$

$$\pi_{il}^k + \sum_{j:(i,j)\in A} c_{ij}x_{ijl}^k \leq R_{\max}, \qquad \forall i \in N, \forall k \in K, l = 1,2 \tag{6.5}$$

$$\sum_{l=1}^{2}(x_{ijl}^k + x_{jil}^k) \leq 1, \qquad \forall \{i,j\} \in E, \forall k \in K \tag{6.6}$$

$$x_{ijl}^k \in \{0,1\}, \qquad \forall k \in K, \forall (i,j) \in A, l = 1,2$$

$$r_i \in \{0,1\}, \qquad \forall i \in V$$

$$\pi_{il}^k \geq 0 \qquad \forall k \in K, \forall i \in V, l = 1,2$$

The objective function (6.2) simply minimizes the total number of regenerators to be placed. Constraints (6.3) are the classical flow conservation constraints from node s_k to node t_k for both paths. Constraint (6.4) linearizes inequality (6.1). Constraint (6.5) enforces that the length of any transparent segment is within the given limit and Constraint (6.6) forces the paths to be edge-disjoint.

A similar model is formulated in [49] where the authors use it only to check feasibility assuming that all the r_i values are fixed.

The results we obtained with this model in our preliminary experimentation did not seem promising. Therefore we decided to apply projection to develop a different formulation.

6.1.1 Projected Formulation

We want to project out π variables from the formulation. So we first assume other variables are fixed. In this case the model can be decomposed for each commodity k and path l. So for a fixed set of variables x and r, and a given k and l, there exists a vector π_l^k that satisfies

$$\begin{aligned}
&\pi_{jl}^k - \pi_{il}^k \geq c_{ij}x_{ijl}^k - R_{\max}(1 - x_{ijl}^k) - R_{\max}r_j && (\gamma_{ij}), && \forall (i,j) \in A,\\
&-\pi_{il}^k \geq \sum_{j:(i,j)\in A} c_{ij}x_{ijl}^k - R_{\max} && (\delta_i), && \forall i \in N,\\
&\pi_{il}^k \geq 0 && && \forall i \in N.
\end{aligned}$$

if and only if

$$\sum_{(i,j)\in A} (c_{ij}x_{ijl}^k - R_{\max}(1 - x_{ijl}^k) - R_{\max}r_j)\gamma_{ij} + \sum_{i\in N}(\sum_{j:(i,j)\in A} c_{ij}x_{ijl}^k - R_{\max})\delta_i \leq 0$$

for all γ and δ that satisfy

$$\sum_{j:(j,i)\in A} \gamma_{ji} - \sum_{j:(i,j)\in A} \gamma_{ij} - \delta_i \leq 0, \qquad \forall i \in N, \tag{6.7}$$

$$\gamma_{ij} \geq 0, \qquad \forall (i,j) \in A, \tag{6.8}$$

$$\delta_i \geq 0, \qquad \forall i \in N. \tag{6.9}$$

If the extreme rays of this projection cone can be identified then an equivalent formulation can be found by means of projection. Fortunately it can be shown that the extreme rays of this polyhedron are either directed paths or directed cycles and the following theorem formalizes this result.

Theorem 6.1 *A vector (γ, δ) is an extreme ray of polyhedron $\mathcal{F} = \{(\gamma, \delta) : (\gamma, \delta)$ satisfies (6.7) and (6.8)$\}$ if the nonzero components of (γ, δ) are all equal in magnitude and form either a simple directed cycle or a simple directed path.*

Proof Considering the inequalities (6.7) and (6.8) it can be seen that if $\gamma_{ij} > 0$ for some $(i, j) \in A$ then $\delta_j + \gamma_{jk}$ must be positive for some $(j, k) \in A$. This means that the support graph induced by γ is composed of directed cycles and directed paths.

First consider a vector (γ, δ) and suppose that the support graph induced by γ consists of only a simple directed cycle $\mathcal{C}$. If $\delta = 0$ we can see that $\gamma_{ij} = \gamma'$ for some $\gamma' \in \mathcal{R}$ for every $(i, j) \in \mathcal{C}$. To find two different rays such that their convex combination includes (γ, δ) we need to choose $\delta = 0$ and note that positive components of γ can be increased or decreased together with the same amount to satisfy (6.7) and (6.8). The vectors obtained this way are simply positive multiples of the initial vector meaning that (γ, δ) is an extreme ray.

Now let $v = (\gamma, \delta)$ be a vector, which induces a support graph consisting of a simple directed cycle and at least another cycle or path. Let $\mathcal{C}$ be the simple cycle, then $\gamma' = \min_{(i,j)\in\mathcal{C}}\{\gamma_{ij}\} > 0$. Clearly, there exists some ϵ such that $\gamma' > \epsilon > 0$. Now consider the vectors $v_1 = (\overline{\gamma}, \overline{\delta})$ ($v_2 = (\underline{\gamma}, \underline{\delta})$) obtained as follows:

$$\overline{\gamma}_{ij} = \begin{cases} \gamma_{ij} + \epsilon, & \text{if } (i,j) \in \mathcal{C} \\ \gamma_{ij}, & \text{otherwise} \end{cases}$$

$$\underline{\gamma}_{ij} = \begin{cases} \gamma_{ij} - \epsilon, & \text{if } (i,j) \in \mathcal{C} \\ \gamma_{ij}, & \text{otherwise} \end{cases}$$

The δ values do not change and $\overline{\delta} = \underline{\delta} = \delta$. It can be easily seen that both v_1 and v_2 are in $\mathcal{F}$ and $v = \frac{v_1+v_2}{2}$. Moreover, neither v_1 nor v_2 is a multiple of v since we do not change all components of v. This implies that v is not an extreme ray of $\mathcal{F}$ and we can conclude if there is a directed cycle in the support graph induced by an extreme ray there can be no other cycles or paths in this ray.

Similar reasoning is valid for the directed paths. Let $v = (\gamma, \delta) \in \mathcal{F}$ be a vector that induces a network including at least one directed path, say $\mathcal{P}$, that ends at node $t \in V$. Without loss of generality we can assume there is no directed cycle in this network. Consider the vectors $v_1 = (\overline{\gamma}, \overline{\delta})$ ($v_2 = (\underline{\gamma}, \underline{\delta})$) obtained as follows:

$$\overline{\gamma}_{ij} = \begin{cases} \gamma_{ij} + \epsilon, & \text{if } (i,j) \in \mathcal{P} \\ \gamma_{ij}, & \text{otherwise} \end{cases}$$

$$\overline{\delta}_{i} = \begin{cases} \delta_{i} + \epsilon, & \text{if } i = t \\ \delta_{i}, & \text{otherwise} \end{cases}$$

$$\underline{\gamma}_{ij} = \begin{cases} \gamma_{ij} - \epsilon, & \text{if } (i,j) \in \mathcal{P} \\ \gamma_{ij}, & \text{otherwise} \end{cases}$$

$$\underline{\delta}_{i} = \begin{cases} \delta_{i} - \epsilon, & \text{if } i = t \\ \delta_{i}, & \text{otherwise} \end{cases}$$

where $\epsilon < \min_{(i,j)\in\mathcal{P}}\{\gamma_{ij}\}$. We can observe that both v_1 and v_2 are in $\mathcal{F}$ and $v = \frac{v_1+v_2}{2}$. Since, neither v_1 nor v_2 is a multiple of (γ, δ), v cannot be an extreme ray of $\mathcal{F}$. This implies that if there is a directed path in an extreme ray there can be no other cycles or paths in this ray.

Based on the results obtained for directed cycles and directed paths, we can conclude that no δ_i can be positive if there is a directed cycle or a directed path in the support graph unless i is the last node of the directed path. This is because we can obtain two different vectors, convex combination of which includes the initial vector, just by increasing and decreasing δ_i by ϵ.

There are two possible cases, when there is only one simple path, $\mathcal{P}$, in the support graph. If the positive components of the vector $v = (\gamma, \delta)$ inducing the support graph have different values, then there exists an ϵ such that $\epsilon < \min_{(i,j)\in\mathcal{P}}\{\gamma_{ij}\}$. Adding and subtracting ϵ to and from v yield two vectors v_1, v_2 such that $v = \frac{v_1+v_2}{2}$. This shows that v is not an extreme ray. On the other hand, if all positive components are equal, the vector is an extreme ray, since it is not possible to increase and decrease a subset of the positive components.

Finally, we need to consider the vector when $\gamma = 0$. Clearly, if there is more than one positive component of δ the vector is not an extreme ray. However, in the case of a single positive δ we have an extreme ray.

Therefore we can conclude that a vector in $\mathcal{F}$ is an extreme ray only if it induces a simple directed cycle or a simple directed path with all positive components are equal or it has a single positive component of δ. $\square$

Now it is known that all extreme rays of the formulation are either directed simple paths or directed simple cycles. The sets of all directed simple paths and directed simple cycles are denoted by $\mathcal{P}$ and $\mathcal{C}$, respectively. Note that this formulation must not be unbounded since our initial mathematical model has a non empty feasible solution set. Therefore, the extreme rays of this formulation must have nonpositive objective function values. This yields three classes of inequalities. Projecting out variables π, we obtain the following formulation denoted by M2. In this formulation, we use $N(P)$ to denote the nodes on $P \in \mathcal{P}$ and p_s, p_t represent the first and the last nodes of P, respectively. Similarly, $N(C)$ stands for the nodes of a cycle $C \in \mathcal{C}$ to denote the nodes

$$\text{minimize} \sum_{i \in N} r_i$$

subject to:

$$\sum_{j:(i,j)\in A} x_{ijl}^k - \sum_{j:(j,i)\in A} x_{jil}^k = \begin{cases} 1, & i = s_k; \\ -1, & i = t_k; \\ 0, & i \neq s_k, t_k; \end{cases} \qquad \forall i \in N, \forall k \in K, l = 1, 2$$

$$\sum_{l=1}^{2} (x_{ijl}^k + x_{jil}^k) \leq 1, \qquad \forall \{i,j\} \in E, \forall k \in K$$

$$\sum_{(i,j)\in P} (R_{\max} + c_{ij}) x_{ijl}^k + \sum_{(p_t,j)\in A} c_{p_t j} x_{p_t jl}^k \leq \sum_{j \in N(P)\setminus\{p_s\}} R_{\max} r_j + (|P|+1) R_{\max}, \qquad \forall k \in K, \forall P \in \mathcal{P}, l = 1, 2 \tag{6.10}$$

$$\sum_{(i,j)\in C} (R_{\max} + c_{ij}) x_{ijl}^k \leq \sum_{j\in N(C)} R_{\max} r_j + |C| R_{\max}, \qquad \forall k \in K, \forall C \in \mathcal{C}, l = 1, 2 \tag{6.11}$$

$$\sum_{j:(i,j)\in A} c_{ij} x_{ijl}^k \leq R_{\max}, \qquad \forall k \in K, \forall i \in N, l = 1, 2 \tag{6.12}$$

$$x_{ijl}^k \in \{0, 1\}, \qquad \forall k \in K, \ \forall (i,j) \in A, \ l = 1, 2$$

$$r_i \in \{0, 1\}, \qquad \forall i \in V$$

In this projected formulation, the objective function and the first two constraints are the same with the original formulation. The objective minimizes the number of regenerators while the first two constraints construct two arc disjoint paths between the source and sink nodes of every commodity. Constraint (6.10) relates the length of paths to the locations of the regenerators. So this constraint is important in the determination of regenerator locations. To explain how this constraint works let $k \in K$ and $u, v \in V$. Consider a path $P \in \mathcal{P}$ between nodes u and v and the associated constraint (6.10). Note that nodes u, v do not have to be the source and sink nodes of commodity k as the constraint must be included for every path and/or path segment regardless of their first and last nodes. Suppose

P is used for commodity k meaning that $\sum_{(i,j)\in P} x_{ijl}^k = |P|$. There are two cases here, P may continue after v if $v \neq t_k$ and otherwise P ends at node v, since the sink node of the commodity is reached. We first analyze the first case, and this implies that there exists a node $w \in V$ such that $(p_t, w) \in A$. So we know that P is extended with the arc (p_t, w). Clearly we have $\sum_{(i,j)\in P} x_{ijl}^k + x_{p_t wl}^k = |P|+1$. As a result the left hand side of the inequality is equal to $\sum_{(i,j)\in P} c_{ij} + c_{p_t w} + |P|R_{\max}$ while the right hand side equals $(|P|+1)R_{\max}$. Replacing x_{ijl} with 1 for every $(i,j) \in P \cup \{(t,w)\}$ and simplifying the terms on both sides we get following inequality:

$$\sum_{(i,j)\in P} c_{ij} + c_{p_t w} \leq \sum_{j\in N(P)\setminus\{p_s\}} R_{\max} r_j + R_{\max} \tag{6.13}$$

Inequality (6.13) states that if the length of the path does not exceed the degradation limit $R_{\max}$, then the path is feasible and no regenerators is necessary. Otherwise, sufficiently many regenerators must be placed on the interior of P so that the exceeded signal degradation is balanced. Note that the first node of P, p_s and the last node of the extended path w are excluded on the right hand side, since a regenerator on these nodes will not have any effect on the infeasibility of the path. Unfortunately, this constraint does not give much information on the locations where the regenerators should be placed, if necessary, even though it provides a lower bound on the number of regenerators that must be installed. Therefore, including only the path constraints associated with the paths between s_k and t_k is not sufficient for the model. To identify the locations as well as the number of regenerators, this constraint must be included in the model for every $P \in \mathcal{P}$. The second case where $v = t_k$ is similar to the first one. In this case, we do not consider the last arc (p_t, w) and the remaining part is the same.

Constraint (6.12) is a special case of Constraint (6.10) when $P = \emptyset$. This constraint simply limits the total length of arcs emanating from the same node.

Constraint (6.11) is used to prevent cycles. The reasoning behind the cycle constraints is the same with the one of the path constraints. A cycle can be considered a path fragment and this implies the use of regenerators if the length of the path fragment exceeds the degradation limit. Let $C \in \mathcal{C}$ and $k \in K$. Assume that C exists in an integer solution, meaning that $\sum_{(i,j)\in C} x^k_{ijl} = |C|$ and therefore we have $\sum_{(i,j)\in C} c_{ij} + |C|R_{\max}$ on the left hand side while we have $|C|R_{\max}$ on the right hand side as constants. Similarly we replace x_{ijl} with 1 for every $(i,j) \in C$ and simplify the terms on both sides of the cycle inequality. This yields

$$\sum_{(i,j)\in C} c_{ij} \leq \sum_{j\in N(C)} R_{\max} r_j \tag{6.14}$$

Inequality (6.14) states that if there is a cycle then there must be some regenerators on some of the nodes of the cycle. Therefore, some fractional solutions are eliminated from the feasible solution set. The cycle inequality (6.11) is not very strict, however, since a cycle can exist if there are sufficiently many regenerators placed on the cycle to satisfy the regeneration requirements of other commodities. This does not constitute a problem since a solution with cycles is feasible provided that there are at least 2 arc disjoint feasible paths between the source and sink nodes of the commodities.

We define the set of feasible solutions to RPP using M2 as follows: $FS = \{(x,r) \in \{0,1\}^{2|K||A|}\text{x}\{0,1\}^{|N|} : (x,r) \text{ satisfies } (6.3),(6.6),(6.10),(6.11)\}$.

6.1.2 Valid Inequalities

We propose two groups of valid inequalities in this section. The first group of inequalities is based on nodes and impose lower bounds on the regenerator variables of other nodes. On the other hand, the second group of valid inequalities consider a path and impose installation of regenerators on the path if necessary.

6.1.2.1 Node Based Valid Inequalities

The shortest path distances between node pairs provide some insight about the number and locations of some regenerators. Since the all-pairs shortest path problem can be solved very quickly, we can use the shortest path distances to develop some valid inequalities for FS. Let d^*_{ij} denote the shortest path distance between nodes i and j.

Proposition 6.1 *Let* $k \in K, l \in \{1,2\}$ *and* $i \in N\setminus\{s_k,t_k\}$ *such that* $d^*_{s_ki}+d^*_{it_k} > R_{\max}$ *and* $\max\{d^*_{s_ki}, d^*_{it_k}\} \le R_{\max}$. *Let* $N_1 = \{j \in N \setminus \{s_k,t_k,i\} : \min\{d^*_{s_ki} + d^*_{ij}, d^*_{ji} + d^*_{it_k}\} \le R_{\max}\}$ *and* $N_2 = \{j \in N \setminus (N_1 \cup \{s_k,t_k,i\}) : d^*_{ji} \le R_{\max}\}$. *Then*

$$2 \sum_{j\in N:(i,j)\in A} x^k_{ijl} \le 2\sum_{j\in N_1} r_j + 2r_i + \sum_{j\in N_2} r_j \tag{6.15}$$

is valid for FS.

Proof If $\sum_{j\in N:(i,j)\in A} x^k_{ijl} = 0$, the inequality is valid. So we assume $\sum_{j\in N:(i,j)\in A} x^k_{ijl} = 1$, which means that i is used on a path from node s_k to node t_k. As $d^*_{s_ki} + d^*_{it_k} > R_{max}$, at least one regenerator is necessary for this path. Since $d^*_{s_ki} \le R_{\max}$ and $d^*_{it_k} \le R_{\max}$, it is possible to find a feasible path from s_k to t_k via i if there is regenerator on i. So the inequality is satisfied if $r_i = 1$. Now assume that $r_i = 0$, then it is clear that i is not an endpoint of a transparent segment. Let u, v be the endpoints of the transparent segment enclosing node i. If $u \in \{s_k,t_k\}$, then $v \notin \{s_k,t_k\}$, and this implies that $r_v = 1$. Since there is a transparent segment between u and v via i, we must have $d^*_{ui} + d^*_{iv} \le R_{max}$ and this means $v \in N_1$. So we have $\sum_{j\in N_1} r_j \ge 1$ and the inequality is satisfied. If $\sum_{j\in N_1} r_j = 0$, then at least two regenerators, which can be reached from i without any regenerators, around i are necessary. This allows construction of a transparent segment enclosing node i inside. Therefore, $\sum_{j\in N_2} r_j \ge 2$. Therefore, inequality (6.15) is valid for FS. □

Proposition 6.2 *Let* $k \in K, l \in \{1,2\}$ *and* $i \in N\setminus\{s_k,t_k\}$ *such that* $d^*_{s_ki} \le R_{\max}$ *and* $d^*_{it_k} > R_{\max}$. *Let* $N_1 = \{j \in N \setminus \{s_k,t_k,i\} : d^*_{s_ki} + d^*_{ij} \le R_{\max}\}$, $N_2 = \{j \in$

$N \setminus (N_1 \cup \{s_k, t_k, i\}) : d^*_{ij} \leq R_{\max}\}$. Then

$$2 \sum_{j \in N:(i,j) \in A} x^k_{ijl} \leq 2 \sum_{j \in N_1} r_j + \sum_{j \in N_2} r_j + r_i \tag{6.16}$$

is valid for FS.

Proof If $\sum_{j \in N:(i,j) \in A} x^k_{ijl} = 0$, the inequality is valid. So we assume $\sum_{j \in N:(i,j) \in A} x^k_{ijl} = 1$, meaning that i is used on a path from node s_k to node t_k. As $d^*_{s_k i} + d^*_{i t_k} > R_{max}$, at least one regenerator is necessary for this path. If there is a regenerator on some node $j \in N_1$, then there may be a transparent segment between s_k and j as $d^*_{s_k i} + d^*_{ij} \leq R_{\max}$ and in this case the inequality is satisfied. So we assume $\sum_{j \in N_1} r_j = 0$. If $r_i = 1$, then two transparent segments connect at i since $i \notin \{s_k, t_k\}$. One transparent segment may start at s_k and end at i while the second one starts at i and ends at some $j \in N_2$. In this case $r_i = r_j = 1$ and the inequality is satisfied. So we assume $r_i = 0$ which makes i an intermediate node of some transparent segment with endpoints u, v. Note that $s_k \notin \{u, v\}$ because of the length constraint for a transparent segment. This implies that $\sum_{j \in N_2} r_j \geq 2$ and the inequality is satisfied. Therefore, inequality (6.16) is valid for FS. □

Proposition 6.3 *Let $k \in K, l \in \{1, 2\}$ and $i \in N \setminus \{s_k, t_k\}$ such that $d^*_{s_k i} > R_{\max}$ and $d^*_{i t_k} \leq R_{\max}$. Let $N_1 = \{j \in N \setminus \{s_k, t_k, i\} : d^*_{ji} + d^*_{i t_k} \leq R_{\max}\}$, $N_2 = \{j \in N \setminus (N_1 \cup \{s_k, t_k, i\}) : d^*_{ij} \leq R_{\max}\}$. Then*

$$2 \sum_{j \in N:(i,j) \in A} x^k_{ijl} \leq 2 \sum_{j \in N_1} r_j + \sum_{j \in N_2} r_j + r_i \tag{6.17}$$

is valid for FS.

Proof Similar to the proof of Proposition 6.2. □

Proposition 6.4 *Let $k \in K, l \in \{1, 2\}$ and $i \in N \setminus \{s_k, t_k\}$ such that $d^*_{s_k i} > R_{\max}$ and $d^*_{i t_k} > R_{\max}$. Let $N_1 = \{j \in N \setminus \{s_k, t_k, i\} : d^*_{ji} \leq R_{\max}\}$. Then*

$$2 \sum_{j \in N:(i,j) \in A} x^k_{ijl} \leq \sum_{j \in N_1} r_j \tag{6.18}$$

is valid for FS.

Proof If $\sum_{j\in N:(i,j)\in A} x^k_{ijl} = 0$, the inequality is valid. So we assume $\sum_{j\in N:(i,j)\in A} x^k_{ijl} = 1$, which means that i is used on a path from node s_k to node t_k. As $d^*_{s_k i} + d^*_{it_k} > R_{max}$, at least one regenerator is necessary for this path. Note that s_k and t_k cannot be endpoints of transparent segments which starts or ends at i. So there must be at least two regenerators which can be reached from i without regenerators, i.e., $\sum_{j\in N_1} r_j \geq 2$, regardless of the regenerator status of node i. Therefore, inequality (6.18) is valid for FS. □

Proposition 6.5 *Let $l \in \{1,2\}$ and $k \in K$ such that $d^*_{s_k t_k} > R_{\max}$. Let $N_1 = \{j \in N \setminus \{s_k, t_k\} : d^*_{s_k j} \leq R_{\max}, d^*_{jt_k} \leq R_{\max}\}$, and $N_2 = \{j \in N \setminus (N_1 \cup \{s_k, t_k\}) : d^*_{s_k j} \leq R_{\max}$ or $d^*_{jt_k} \leq R_{\max}\}$. Then*

$$2\sum_{j\in N_1} r_j + \sum_{j\in N_2} r_j \geq 2 \tag{6.19}$$

is valid for FS.

Proof As $d_{s_k t_k} > R_{max}$ at least one regenerator is necessary for this pair. If there is a node j such that $d_{s_k j} \leq R_{max}$ and $d_{jt_k} \leq R_{max}$, which means $j \in N_1$, then one regenerator at node j is sufficient to make this pair feasible. Now suppose $r_j = 0$ for every $j \in N_1$. Clearly, we have more than two transparent segments but we are interested in two of them, in particular the one that starts from s_k and the one that ends at t_k. Let u, v be the nodes which are endpoints of these transparent segments, respectively. Clearly, $u, v \in N_2$ and $\sum_{j\in N_2} r_j \geq 2$. Therefore, inequality (6.19) is valid for FS. □

Note that the valid inequalities (6.15)-(6.19) can be improved by introducing some auxiliary variables to the formulation. These auxiliary variables can be used to force two or three r variables to be equal to 1 simultaneously. However, this will increase the size of the formulation, which is already large. Additionally, our initial experimentation showed that the improvement in the lower bounds is not much. So we decided not to include the auxiliary variables in the formulation.

6.1.2.2 Path Based Valid Inequalities

We also propose some logical valid inequalities based on paths. These inequalities are similar to Constraint (6.10), however the arc lengths are not included in the inequalities. Instead, the paths are chosen such that they satisfy some particular conditions. This allows us to make Constraint (6.10) stronger under some conditions. Let $l \in \{1,2\}, k \in K$ and P be a path with $L(P) \leq R_{max}$. Clearly, if the signal is regenerated at the beginning of this path then no regeneration is needed along the path. Suppose that a new arc is added at the end of this path. If the length of the path is still less than degradation limit there is no need for regenerators. However, if the addition of the last arc yields a path with length longer than $R_{\max}$, then clearly the signal must be regenerated on P, therefore a regenerator must be placed on at least one of the intermediate nodes of this path. So we are interested in arcs, whose addition makes regeneration necessary. We use N_+ to denote the set of such arcs. Similarly, the arcs can be added to the front of the path instead of the end and N_- denotes this set. We formally define both sets as follows:

- $N_+(P) = \{j \in N \setminus N(P) : (p_t, j) \in A, L(P) + c_{p_t j} > R_{max}\}$,
- $N_-(P) = \{j \in N \setminus N(P) : (j, p_s) \in A, c_{j p_s} + L(P) > R_{max}\}$.

For ease of notation, we write $x_l^k(P)$ instead of $\sum_{(i,j)\in P} x_{ijl}^k$. Based on this information we now can propose a valid inequality.

Lemma 6.1 *Let $l \in \{1,2\}, k \in K$ and P be a directed path with $L(P) \leq R_{max}$. Then*

$$x_l^k(P) + \sum_{j \in N_+(P)} x_{p_t j l}^k \leq |P| + \sum_{j \in N(P)\setminus\{p_s\}} r_j \tag{6.20}$$

is valid for FS and dominates inequality (6.10).

Proof Note that, inequality (6.20) is violated if the left hand side is equal to $|P| + 1$, i.e., a path segment composed of P and an arc from $N_+(P)$ is used.

So without loss of generality we can assume $N_+(P) \neq \emptyset$. This results in a path segment longer than $R_{\max}$ which makes a regenerator on at least one of the interior nodes necessary. Therefore inequality (6.20) is valid for FS. Now we want to compare it with inequality (6.10) for the same P.

We can rewrite inequality (6.10) as $\sum_{(i,j)\in P}(R_{\max}+c_{ij})x^k_{ijl}+\sum_{j\in N_+} c_{p_tj}x^k_{p_tjl}+\sum_{j\in N\setminus N_+:(p_t,j)\in A} c_{p_tj}x^k_{p_tjl} \le \sum_{j\in N(P)\setminus\{p_s\}} R_{\max}r_j + (|P|+1)R_{\max}$. Dividing each term by $R_{\max}$ and rearranging the inequality we obtain,

$$\begin{aligned} x^k_l(P) + \sum_{j\in N_+} \frac{c_{p_tj}}{R_{\max}} x^k_{p_tjl} \le |P| + \sum_{j\in N(P)\setminus\{p_s\}} r_j \\ + (1 - \sum_{(i,j)\in P} \frac{c_{ij}}{R_{\max}} x^k_{ijl} - \sum_{j\in N\setminus N_+:(p_t,j)\in A} \frac{c_{p_tj}}{R_{\max}} x^k_{p_tjl}) \end{aligned}$$

It can be seen that the term in parenthesis is nonnegative. So we can say that the left (right) hand side of inequality (6.10) is less (greater) than or equal to that of inequality (6.20). Therefore (6.10) is dominated by (6.20). □

Since the graph is directed there is another path visiting the same nodes in reverse order. For some path P, $\overleftarrow{P}$ stands for the path which is in the opposite direction of P. In the following proposition we show how inequality (6.20) is modified to make it stronger.

Proposition 6.6 *Let $l \in \{1,2\}, k \in K$ and P be a directed path with $L(P) \le R_{max}$. Then*

$$x^k_l(P) + x^k_l(\overleftarrow{P}) + \sum_{j\in N(P):(p_t,j)\in A\setminus\overleftarrow{P}} x^k_{p_tjl} + \sum_{j\in N_+(P)\setminus N(P)} x^k_{p_tjl} \le |P| + \sum_{j\in N(P)\setminus\{p_s\}} r_j \tag{6.21}$$

is valid for FS.

Proof We know that $x^k_l(P) + \sum_{j\in N_+(P)\setminus N(P)} x^k_{p_tjl} \le |P| + \sum_{j\in N(P)\setminus\{p_s\}} r_j$ is valid for FS from Lemma 6.1. So we analyze the newly added terms. Let

i_1, i_2, i_3, i_4 be four consecutive nodes on P and assume that $x^k_{i_3i_2l} = 1$. This implies $x^k_{i_2i_3l} = x^k_{i_1i_2l} = 0$. In other words, if the x variable for an arc on $\overleftarrow{P}$ is equal to 1, then two x variables corresponding to arcs on P are forced to be 0. Consequently, $x^k_l(P) + x^k_l(\overleftarrow{P}) + \sum_{j\in N_+(P)\setminus N(P)} x^k_{p_tjl} \leq |P|$, and the inequality is satisfied. The case $\sum_{j\in N(P):(p_t,j)\in A\setminus\overleftarrow{P}} x^k_{p_tjl} = 1$ is similar. So we assume that $x^k_l(\overleftarrow{P}) + \sum_{j\in N(P):(p_t,j)\in A\setminus\overleftarrow{P}} x^k_{p_tjl} = 0$ and the remaining part is shown to be valid for FS. Therefore inequality (6.21) is valid for FS. □

Clearly, inequality (6.21) is a lifted version of (6.20) and dominates it. In addition, we can find a similar valid inequality using N_- as follows:

Proposition 6.7 *Let $l \in \{1,2\}, k \in K$ and P be a directed path with $L(P) \leq R_{max}$. Then*

$$x^k_l(P) + x^k_l(\overleftarrow{P}) + \sum_{j\in N(P):(p_s,j)\in A\setminus P} x^k_{p_sjl} + \sum_{j\in N_-(P)\setminus N(P)} x^k_{jp_sl} \leq |P| + \sum_{j\in N(P)\setminus\{p_t\}} r_j \tag{6.22}$$

is valid for FS.

Proof Similar to the proof of Proposition 6.6. □

We end this section by presenting two more classes of valid inequalities based on paths.

Proposition 6.8 *Let $l \in \{1,2\}, k \in K$ and P be a directed path with $L(P) \leq R_{max}$, $p_t \neq t_k$, and $N_+(P) \neq \emptyset$. Then*

$$x^k_l(P) \leq |P| - 1 + \sum_{j\in N(P)\setminus\{p_s\}} r_j + \sum_{j\in N\setminus N_+(P):(p_t,j)\in A} x^k_{p_tjl} \tag{6.23}$$

is valid for FS.

Proof If $x^k_l(P) \leq |P| - 1$ then the inequality is valid. So we assume $x^k_l(P) = |P|$ meaning that our route from s_k to t_k uses the path segment from p_s to p_t. As

$p_t \neq t_k$, there are two alternatives. Either the path continues through an arc (p_t, j) which satisfies $L(P) + c_{p_t j} \leq R_{max}$, or the addition of the new arc makes it necessary to have a regenerator on an intermediate node as the path length exceeds R_{max}. Therefore inequality (6.23) is valid for FS. □

Proposition 6.9 *Let $l \in \{1,2\}, k \in K$ and P be a directed path with $L(P) \leq R_{max}$, $p_s \neq s_k$, and $N_-(P) \neq \emptyset$. Then*

$$x_l^k(P) \leq |P| - 1 + \sum_{j \in N(P)\setminus\{p_t\}} r_j + \sum_{j \in N\setminus N_-(P):(j,p_s)\in A} x_{jp_sl}^d \qquad (6.24)$$

is valid for FS.

Proof Similar to the proof of Proposition 6.8. □

6.2 Solving M2

Let $K' \subseteq K$ be a commodity set. M2 can be used to find the optimal solution of RPP for K'. However, as the number of constraints of M2 is exponential, a branch and cut algorithm is necessary. In this section we propose a branch and cut algorithm to solve M2. Before the implementation details of the branch and cut algorithm, we want to discuss the separation problems associated with constraints (6.10) and (6.11) and valid inequalities (6.21)-(6.24) which will be added to the model during the optimization. Note that all constraints and valid inequalities are associated with a specific commodity k and path type l, that is to say we do not have an inequality that includes different commodities or path types (working and restoration paths). This allows us to solve the separation problems for each $k \in K'$ and $l = 1, 2$. So for a given fractional solution $(\overline{x}, \overline{r})$, fixed $k \in K'$ and $l \in \{1,2\}$ we define the following node and arc sets: $N_{kl}^* = \{i \in N : \exists j \in N \text{ such that } \overline{x}_{ijl}^k + \overline{x}_{jil}^k > 0\}$, $A_{kl}^* = \{(i,j) \in A : \overline{x}_{ijl}^k > 0\}$, and $R^* = \{i \in N : \overline{r}_i > 0\}$. Now we can define our support graph $D^{kl} = (N_{kl}^*, A_{kl}^*)$. Separation procedure is performed in two phases, we first separate inequalities (6.10) and (6.11) exactly and then apply heuristic separation for inequalities (6.22)-(6.24).

6.2.1 Exact Separation

This part is for the exact separation associated with inequalities (6.10) and (6.11). Rearranging constraint (6.10) we get the following inequality:

$$\begin{aligned}&\Big(\sum_{j\in N(P)\setminus\{p_s\}} R_{\max} r_j + |P|R_{\max} - \sum_{(i,j)\in P}(R_{\max}+c_{ij})x^k_{ijl}\Big)\\ &+ \Big(R_{\max} - \sum_{(p_t,j)\in A} c_{p_tj}x^k_{p_tjl}\Big)\end{aligned} \geq 0$$

In order to get a simpler representation, we let

$$\begin{aligned}\alpha &= \sum_{j\in N(P)\setminus\{p_s\}} R_{\max} r_j + |P|R_{\max} - \sum_{(i,j)\in P}(R_{\max}+c_{ij})x^k_{ijl}\\ \beta &= R_{\max} - \sum_{(p_t,j)\in A} c_{p_tj}x^k_{p_tjl}\end{aligned}$$

so that constraint (6.10) becomes $\alpha+\beta \geq 0$. Clearly, our aim is to minimize $\alpha+\beta$ to identify if there is a violated inequality. Since the constraint is defined by a directed path, the values of α and β should depend on the values of the variables corresponding to the arcs on the path. This is true for α, however, β consists of variables corresponding to arcs that emanate from a single node, which is the last node of the path. In other words, β does not depend on the arcs on the path, and its value is just determined by the last node of a path. This means that different paths that end at the same node will have the same β value. This allows us to break the separation problem into smaller parts for each node and fortunately the separation problem reduces to the minimization of α for a given node.

Now we show how α can be minimized. Consider the support graph $D^{kl} = (N^*_{kl}, A^*_{kl})$ we defined and set the weight of each arc $(i,j) \in A^*_{kl}$ to $w_{ij} = R_{\max}(1+\overline{r}_j - \overline{x}^k_{ijl}) - c_{ij}\overline{x}^k_{ijl}$. Let P be a directed path on this graph. It can be easily verified that the length of P on D^{kl} is equal to α. Therefore for a given node pair s,t the shortest path between s and t gives the minimum α value. As β is known for the last node t, regardless of the path, it can be determined if there exists a path P

between s, t which yields a violated path inequality by finding the shortest path problem between s and t.

Let α_{ij} denote the minimum α value between nodes i and j. Similarly let β_i denote the β value associated with node i. Solving an all-pairs shortest path problem on D^{kl} we can find the α_{st} for every node pair s and t. Now it can be seen whether there is a violated path inequality induced by a path between nodes s and t by checking if $\alpha_{st} + \beta_t < 0$.

It is known that shortest path problem can be solved in polynomial time if there is not any negative directed cycle on the network. So we also need to analyze the case with negative directed cycles. Suppose a negative cycle, C, is found while solving the shortest path problem. This means that $\sum_{j \in N(C)} R_{\max} r_j + |C| R_{\max} < \sum_{(i,j) \in C} (R_{\max} + c_{ij}) x^k_{ijl}$ and hence inequality (6.11) is violated.

Therefore by solving the shortest path problem we not only find the violated path constraints (6.10), but also identify violated cycle inequalities (6.11) if there is any. This shows that the separation problems of Constraints (6.10) and (6.11) can be solved together. A label correcting shortest path algorithm (see [1]) is used to solve the all-pairs shortest path problem. This algorithm either solves the shortest path problem or detects a negative cycle in polynomial time. So we can conclude that if there is a violated inequality of type (6.10) or (6.11), it can be found in polynomial time.

At the end of the algorithm if we encounter a negative directed cycle then corresponding inequality of type (6.11) is added. If there is no negative directed cycle, then for every node $t \in N$ we choose the node $s \in N \setminus \{t\}$ that gives the minimum α_{st} which is closest to t and check if the path between these two nodes induce a violated inequality of type (6.10). Remember that inequality (6.21) dominates inequality (6.10) if the length of the path is smaller than the degradation limit. Therefore, in case of a violated inequality, we add inequality (6.21) if the length of path does not exceed $R_{\max}$ and add inequality (6.10), otherwise. Exact separation algorithm is described in Algorithm 9.

Algorithm 9: Exact Separation

```
   Input: $(\overline{x}, \overline{r}); K$
1  forall $k \in K$ do
2      forall $l \in \{1, 2\}$ do
3          Construct $D^{kl} = (N^*_{kl}, A^*_{kl})$
4          forall $(i, j) \in A^*_{kl}$ do
5              $w_{ji} \leftarrow R_{\max}(1 + \overline{r}_j - \overline{x}^k_{ijl}) - c_{ij}\overline{x}^k_{ijl}$
6          forall $i \in N^*_{kl}$ do
7              Find shortest paths from $i$ to every $j \in N^*_{kl} \setminus \{i\}$ $(d_{ij})$
8              if There is a negative cycle $C$ then
9                  Add inequality (6.11) corresponding to $C$
10             else
11                 $t \leftarrow \underset{j \in N^*_{kl} \setminus \{i\}}{\operatorname{argmin}} \{d_{ij}\}$
12                 if $d_{it} + R_{\max} - \sum_{(i,j) \in A} c_{ij}\overline{x}^k_{ijl} < 0$ then
13                     There is violated inequality of type (6.10)
14                     Add inequality (6.10) or (6.21) corresponding to the
                       path between $i$ and $t$
```

6.2.2 Heuristic Separation

We consider the directed graph D^{kl} and set the weight of each arc $(i, j) \in A^*_{kl}$ to $w_{ij} = \overline{x}^k_{ijl}$. On D^{kl}, we try to find the longest path starting at node s_k. We use a greedy heuristic algorithm to find this path. We start from s_k and at each step we choose the arc which emanates from the last node and ends at a node not visited before. The heuristic stops if t_k is visited or an arc cannot be found. This path is used for the separation of inequalities (6.22)-(6.24), as follows. We start from each node of this path and add the arcs in the order of their place on the path. When the length of the path segment exceeds $R_{\max}$, this path segment is a candidate. We check if an inequality of type (6.21)-(6.24) is violated. If there is a violation then the violated inequalities are added to the subproblem. Heuristic separation algorithm is described in Algorithm 10.

Algorithm 10: Heuristic Separation

```
   Input: $(\overline{x}, \overline{r}); K$
1  forall $k \in K$ do
2      forall $l \in \{1, 2\}$ do
3          Construct $D^{kl} = (N^*_{kl}, A^*_{kl})$
4          forall $(i, j) \in A^*_{kl}$ do
5              $w_{ij} \leftarrow \overline{x}^k_{ijl}$
6          $P \leftarrow \emptyset, v \leftarrow s_k$
7          repeat
8              $T \leftarrow \{w_{vj} : j \in N^*_{kl} \setminus N(P),\ (v, j) \in A^*_{kl}\}$
9              if $T \neq \emptyset$ then
10                 $(v, u) \leftarrow \underset{j \in N^*_{kl} \setminus N(P) : (v,j) \in A^*_{kl}}{\text{argmax}} \{w_{vj}\}$
11                 $P \leftarrow P \cup \{(v, u)\}$
12                 $v \leftarrow u$
13         until $v = t_k$ or $T = \emptyset$ ;
14         forall $s \in N(P)$ do
15             Find the longest segment of $P$, say $P'$, starting at $s$ and not
               exceeding $R_{\max}$
16             if $P'$ violates one of the inequalities of type (6.21)-(6.24) then
17                 Add corresponding violated inequalities of type (6.21)-(6.24)
                   induced by $P'$
```

6.3 Finding the Optimal Solution

Note that as $|K|$ increases the size of M2 also increases. This makes solving M2 for all commodities simultaneously almost impossible. Therefore, we propose an iterative algorithm to find the optimal solution of RPP. This algorithm first solves the RPP for a restricted subset K' of K, and checks if this solution is feasible for K. New commodities are added to K' until the optimal solution for K' is feasible for K. So the algorithm has two main operations, finding the optimal solution for some $K' \subseteq K$ and checking if a given solution is feasible for K. We first start with a single commodity ($|K'| = 1$) and find its optimal solution using $M2$. Then we check if this solution is feasible for other commodities. If it is feasible then the solution is optimal. Otherwise, we add the infeasible commodity k' to K' and solve M2 for K' again. This procedure is repeated until the optimal solution for K is found. This method is provided in Algorithm 11. Now we want to discuss how feasibility of a commodity with respect to a given regenerator set is analyzed.

Algorithm 11: Optimal Solution

```
   Input: $G = (N, E); K$
1  $K' \leftarrow \emptyset$
2  Choose a $k \in K \setminus K'$
3  $K' \leftarrow K' \cup \{k\}$
4  repeat
5      flag $\leftarrow$ false
6      Solve M2 for $K'$
7      $R \leftarrow R^*$ (regenerator set in the optimal solution of M2 for $K'$)
8      forall $k \in K \setminus K'$ do
9          if $k$ is not feasible with respect to $R$ then
10             flag $\leftarrow$ true
11             $K' \leftarrow K' \cup \{k\}$
12             break
13 until flag=false ;
```

6.3.1 Feasibility Check

M1 can be used to check if a regenerator set R is feasible for a node pair in its current form, however, we will improve it by applying a reduction procedure. Let $k \in K$ be a commodity for which feasibility check is performed, R be the regenerator set, and $i \in N \setminus (R \cup \{s_k, t_k\})$. If the path between s_k, t_k includes i, the length of the transparent segment entering node i will be either equal to $d^*_{s_k i}$ provided no regenerator node is visited before i or d^*_{ji} provided that j is the last regenerator node visited before i. Similarly, the transparent segment will either end up in t_k or in a regenerator node j and its length will be equal to $d^*_{it_k}$ or d^*_{ij}, respectively. Therefore, the following inequalities have to be satisfied if i is on the path, where $\beta^k_i = \min\{d^*_{ij} | j \in (R \cup \{s_k\}) \setminus \{t_k\}\}$ and $\gamma^k_i = \min\{d^*_{ij} | j \in (R \cup \{t_k\}) \setminus \{s_k\}\}$.

$$\pi^k_{il} \leq R_{max} - \gamma^k_i, \qquad \forall i \in N \setminus (R \cup \{t_k\}), \forall k \in K, l = 1, 2 \tag{6.25}$$

$$\pi^k_{il} \geq \beta^k_i, \qquad \forall i \in N \setminus (R \cup \{s_k\}), \forall k \in K, l = 1, 2 \tag{6.26}$$

Inequalities (6.25)-(6.26) impose bounds on the portion of the transparent segment that includes node i. However, note that if $\beta_i + \gamma_i > R_{\max}$ then the upper bound of π^k_{il} is less than the lower bound which results in an inconsistency. This means that there cannot be a transparent segment visiting node i. Therefore, if $\beta_i + \gamma_i > R_{\max}$, node i can be removed from the graph without ignoring any feasible path. Similarly, opaque nodes can also be removed from the network if $\beta_i > R_{\max}$ or $\gamma_i > R_{\max}$ as an opaque node constitutes an end-point for a transparent segment. If we have $\beta_{t_k} > R_{\max}$ or $\gamma_{s_k} > R_{\max}$, it can directly be concluded that k is infeasible with respect to R. Applying the reduction, which is described in Algorithm 12 the number of nodes and edges can be significantly decreased. Moreover, remember that both β and γ values are computed using the shortest path distances of the original graph. However, they need to be updated using the shortest path distances of the reduced graph which means that it may be possible to reduce the graph even more by applying the procedure above after solving a shortest path problem on the reduced graph. Using this reduction, feasibility problem can be solved on a smaller network.

6.4 Computational Results

Based on the separation problems described in Section 6.2 we develop a branch-and-cut algorithm to solve M2. We start the algorithm by solving the following initial linear program.

$$
\begin{aligned}
&\text{minimize } \sum_{i \in N} r_i \\
&\text{subject to:} \\
&(6.3), (6.6), (6.15), (6.16), (6.17), (6.18), (6.19) \qquad && \forall k \in K', l = 1, 2 \\
&0 \le x_{ijl}^k, r_i \le 1, && \forall i, j, k, l
\end{aligned}
$$

If the solution of this problem $(\overline{x}, \overline{r})$ satisfies inequalities (6.10) and (6.11), then it is feasible for RPP. To find if there are any violated inequalities of these type we solve separation problems at each step of the branch-and-cut algorithm. Heuristic separation is performed firstly to find inequalities of type (6.21)-(6.24). If no violated inequality could be found exact separation algorithm is used. We generate at most 200 inequalities at each step.

We used a network with 32 nodes and 50 edges shown in Figure 6.1. This network is based on US data and is used also by Yetginer and Karaşan [49]. $R_{\max}$ values are chosen such that decreasing it, will result in infeasible problems and while increasing we obtain cases where no regeneration is needed. Our algorithm is implemented using C++ with Concert Technology 27 and Cplex 11.2 is used to solve the mathematical models. We run the algorithm on a workstation with 2.67 Ghz Pentium Xeon Cpu and 8 Gb ram.

Using the proposed solution algorithm, the optimal solutions for various $R_{\max}$ values are found in less than 60 minutes. The optimal solutions and Cpu times

Algorithm 12: Reduction

```
   Input: $G = (N, E); k \in K; R \subseteq N$
1  repeat
2      flag← false
3      forall $i \in N$ do
4          if $i = s_k$ then
5              $\beta_i = 0$
6              $\gamma_i = \min\{d^*_{ij} | j \in (R \cup \{t_k\}) \setminus \{s_k\}\}$
7          else if $i = t_k$ then
8              $\gamma_i = 0$
9              $\beta_i = \min\{d^*_{ij} | j \in (R \cup \{s_k\}) \setminus \{t_k\}\}$
10         else
11             $\beta_i = \min\{d^*_{ij} | j \in (R \cup \{s_k\}) \setminus \{i, t_k\}\}$
12             $\gamma_i = \min\{d^*_{ij} | j \in (R \cup \{t_k\}) \setminus \{i, s_k\}\}$
13     if $\beta_{t_k} > R_{\max}$ or $\gamma_{s_k} > R_{\max}$ then
14         $R$ is infeasible; stop
15     forall $i \in N \setminus (R \cup \{s_k, t_k\})$ do
16         if $\beta_i + \gamma_i > R_{\max}$ then
17             $N \leftarrow N \setminus \{i\}$
18             $E \leftarrow E \setminus \delta(i)$
19             flag← true
20     forall $i \in R \setminus \{s_k, t_k\}$ do
21         if $\beta_i > R_{\max}$ or $\gamma_i > R_{\max}$ then
22             $N \leftarrow N \setminus \{i\}$
23             $E \leftarrow E \setminus \delta(i)$
24             $R \leftarrow R \setminus \{i\}$
25             flag← true
26     if flag then
27         Solve shortest path problem on $G = (N, E)$
28 until flag=false ;
```

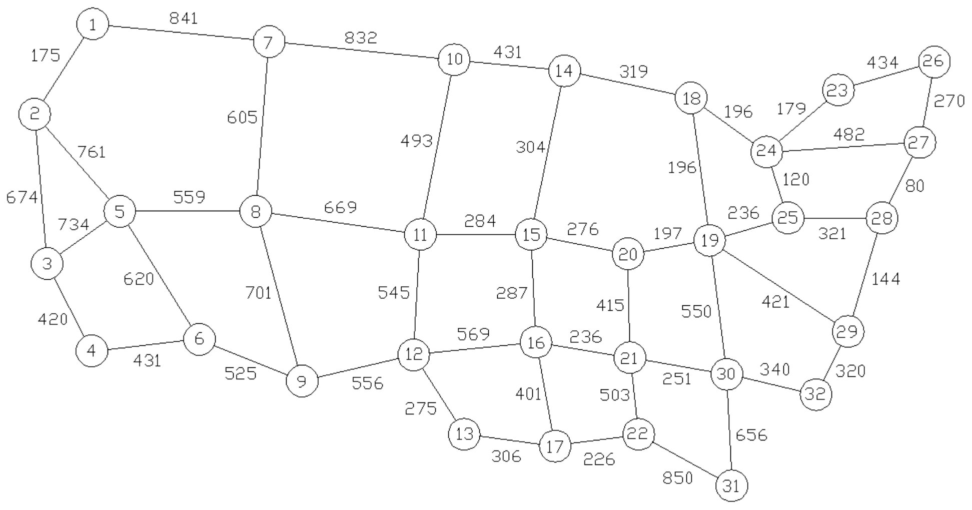

Figure 6.1: 32 node network.

are tabulated in Table 6.1. When the results are analyzed it can be seen that there is no correlation between the $R_{\max}$ values and Cpu times. For example, the problem can be easily solved when $R_{\max} = 1000$ or $R_{\max} = 1200$, while it takes significantly longer to solve the problem with $R_{\max} = 1100$. Similar deviations in Cpu times are also observed for different $R_{\max}$ values.

6.5 Conclusion

In this chapter, the regenerator placement problem is analyzed and a solution methodology for this problem is developed. Using the proposed strategy, the RPP is solved to optimality for various $R_{\max}$ values on a graph with 32 nodes and 50 edges. Although the problem can be solved within reasonable times, the effectiveness of the algorithm decreases as the size of the problem increases since the increase in number of nodes will also result in a larger commodity set. The

Table 6.1: Optimal solutions for the 32 node network problem.

$R_{\max}$	# of regs. placed	Time (s)	$R_{\max}$	# of regs. placed	Time (s)
900	12	61.11	2000	3	337.11
1000	11	49.21	2100	3	480.12
1100	10	3031.11	2200	2	21.86
1200	8	206.95	2300	2	765.97
1300	7	46.66	2400	2	7.43
1400	6	65.33	2500	2	3.79
1500	6	1639.84	2600	2	4.34
1600	5	118.68	2700	2	59.7
1700	4	17.68	2800	1	2.04
1800	4	285.48	2900	1	5.23
1900	3	173.32	3000	1	4.47

current solution methodology attacks the problem using a small subset of commodities and tries to prove optimality by a feasibility checking procedure. There are two problems with this approach. First, many feasibility problems must be solved. Although the size of the feasibility problems can be reduced significantly with the proposed reduction procedure, it still takes significant amount of Cpu time. Second problem is about the forming the commodity subset. The way we choose the commodities which will compose the commodity subset affects the solution times. This subset can be formed in many different ways and unfortunately we could not determine one that works well for all $R_{\max}$ values.

The current formulations make it necessary to start with a small commodity subset, since including all commodities will result in a huge integer program which is almost impossible to solve. Another drawback of the formulations is their weak LP relaxation values, which is equal to 0, regardless of the optimal solution. This clearly makes it very difficult to prove optimality of a solution even if that solution is optimal. The node based valid inequalities improve the LP relaxations, however, it is still very hard to include many commodities in the formulation. Therefore, another mathematical formulation may be useful. A path based integer program may provide better LP relaxations and may allow inclusion of many commodities in the formulation. As such a formulation will include exponential number of variables, a branch-and-price algorithm will be required.

Another future research direction is about the signal degradation. In this chapter, it is assumed that signal degradation is a linear function of the distance traveled. However, a nonlinear signal degradation function will provide a better representation of the real life problems. Including a nonlinear degradation function will change the distance constraints of the formulation, and this new model might be transformed into a conic programming model. A conic program may provide better relaxation values and may decrease the solution times as the performance of commercial conic programming solvers is rapidly improving.

Finally, we need to note that solving the design problem of a backbone network and the RPP on the same backbone network would yield suboptimal solutions. These problems are attacked separately in this dissertation as the RPP is very difficult even if it is solved alone. However, integrating both problems would be a good direction for future research.

Chapter 7

Conclusion

In this dissertation, we focused on some network design and network routing problems as well as location problems which are motivated by the vast development of the internet technologies and rapid increase in the data traffic on the telecommunications networks. The first problem we studied is a two-level survivable network design problem, which is referred to as 2ECSSP. The network we analyze consists of a backbone network, interconnecting the hubs, and local access networks, connecting the users to the backbone network. To achieve survivability, the backbone network is designed as a 2-edge connected network. The local access networks have star architecture, i.e., each user is directly connected to exactly one hub. An integer linear program is formulated for 2ECSSP and some valid inequalities which strengthen the formulation are proposed. The polyhedral structure of the polytope associated with this formulation is analyzed and the conditions under which the constraints and valid inequalities define facets are provided.

The separation problems used to identify violated constraints and/or valid inequalities are described. In addition, some reduction operations are proposed in order to reduce the size of the separation problems that must be solved within the branch-and-cut algorithm. It is shown that applying these reduction operations the size of a fractional solution can be reduced and another fractional solution on a reduced space can be obtained. It is also shown that the violated inequalities of the original solution are preserved in the reduced solution. In other words, separation problems can be solved after reduction operations are applied and the violated inequalities can still be found. A branch-and-cut algorithm is devised to find the optimal solution of the problem, and through computational analysis, the effectiveness of the branch-and-cut algorithm is tested. The computational experiments have shown that the reduction operations could improve the solution performance of the algorithm significantly in some cases. We observed that problem instances with more than 300 nodes could be solved using the proposed methodology.

The survivability is extended to the local access networks in the second problem we analyzed in the dissertation. Each user is connected to two hubs instead of one, as done in 2ECSSP, to achieve further survivability. This problem is referred to as 2ECSDHP due to the dual homing structure in the local access networks. Three integer linear formulations are developed and their strengths are compared. Due to the reasons explained in Chapter 5, the strongest formulation is not selected to be used in computations but some valid inequalities are obtained from this one by the projection method. The polyhedral analysis of the polytope associated with the integer program is performed, and the separation problems of the constraints and valid inequalities are also discussed. To improve the performance of the branch-and-cut algorithm devised to find the optimal solution of the problem some variable fixing rules are proposed. In the computational analysis, we used the branch-and-cut algorithm to find the optimal solutions. We also analyzed the improvement gained by the application of variable fixing rules.

In the third and last problem, namely the regenerator placement problem, we focused on location and network routing problems. A multi-commodity flow based integer program was developed and due to some performance issues another integer linear program with exponential number of constraints was formulated by means of projection. In order to make the formulation stronger some valid inequalities are introduced. The shortest path distances between node pairs are used to define these valid inequalities. Besides, some valid inequalities which dominate the path constraints of the formulation are also proposed. It is difficult to solve this problem directly as there is a large number of commodities.

Therefore an iterative method is proposed to achieve optimality where a subset of commodities are handled at each iteration. A branch-and-cut algorithm is developed to search the optimal solution and using this together with a method that is used to identify whether a solution for a subset of commodities is feasible or not for the entire set of commodities, the optimal solutions are found.

In all problems we analyzed the capacities of the network components are assumed to be infinite. However, in real life applications, the component will have finite capacities. Therefore, incorporating the capacities to the problems will be an interesting variation for the problems. The capacitated versions of single level network design problems are widely studied in the literature. Therefore, the capacitated 2ECSSP and 2ECSDHP will contribute to the literature and hence will be good candidates for future research.

In addition, different survivability requirements may be desired for the networks in practice. Therefore, the two-level network design which use different network architectures can be analyzed. A (1,2) connected or k-connected backbone network where $k > 2$ might be an interesting research as the polyhedral structure significantly changes if k is odd.

Finally the concept of bounded rings can be considered for the backbone network. Clearly, an edge of the backbone can be a part of a quite large cycle which will result in a significantly longer secondary path in case of a failure.

In this dissertation we have studied three problems related to network optimization. These problems have not been studied in the literature yet. Therefore, our study is important in the sense that some variants of problem that arises in the telecommunications context have been thoroughly analyzed. In addition to our contribution to the literature, this study will form a basis for the related future research problems.

Bibliography

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms and Applications*. Prentice-Hall, 1993.

[2] M. Baïou and J. R. Correa. The node-edge weighted 2-edge connected subgraph problem: Linear relaxation, facets and separation. *Discrete Optimization*, 3:123–135, 2006.

[3] M. Baïou and A. R. Mahjoub. Steiner 2-edge connected subgraph polytopes on series-parallel graphs. *SIAM Journal on Discrete Mathematics*, 10:505–514, 1997.

[4] E. Balas and M. W. Padberg. Set partitioning: A survey. *SIAM Rev*, 18:710–760, 1976.

[5] A. Banerjee, J. Drake, J. Lang, B. Turner, D. Awduche, L. Berger, K. Kompella, and Y. Rekhter. Gmpls: An overview of signaling enhancements and recovery techniques. *IEEE Communications Magazine*, 39(7):144–151, July 2001.

[6] F. Barahona and A. R. Mahjoub. On two-connected subgraph polytopes. *Discrete Mathematics*, 147:19–34, 1995.

[7] N. Barakat and A. L. Garcia. An analytical model for predicting the locations and frequencies of 3r regenerations in all-optical wavelength-routed wdm networks. *ICC 2002 - IEEE International Conference on Communications*, 1:2812–2816, April 2002.

[8] M. S. Borella, J. P. Jue, D. Banerjee, B. Ramamurthy, and B. Mukherjee. Optical components for wdm lightwave networks. In *Proc IEEE*, volume 85, pages 1274–1307, 1997.

[9] S. Chen, I. Ljubic, and S. Raghavan. The regenerator location problem. *Networks*, 55:205–220, 2010.

[10] G. Cornuéjols, D. Naddef, and W. R. Pulleyblank. Halin graphs and the travelling salesman problem. *Mathematical Programming*, 26:287–294, 1983.

[11] G. Dahl, D. Huygens, A. R. Mahjoub, and P. Pesneau. On the k edge-disjoint 2-hop-constrained paths polytope. *Operations Research Letters*, 34:577–582, 2006.

[12] J. Fonlupt and A. R. Mahjoub. Critical extreme points of the 2-edge connected spanning subraph problem. *Math Program B*, 105:289–310, 2006.

[13] B. Fortz and M. Labbé. Polyhedral results for two-connected networks with bounded rings. *Mathematical Programming*, 93:27–54, 2002.

[14] B. Fortz, M. Labbé, and F. Maffioli. Solving the two-connected network with bounded meshes problem. *Operations Research*, 48(6):866–877, 2000.

[15] B. Fortz, A. R. Mahjoub, S. T. McCormick, and P. Pesneau. Two-edge connected subgraphs with bounded rings: Polyhedral results and branch-and-cut. *Mathematical Programming*, 105:85–111, 2006.

[16] P. Fouilhoux, O. E. Karaşan, A. R. Mahjoub, O. Özkök, and H. Yaman. Survivability in hierarchical telecommunications networks. *Networks*, To Appear.

[17] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum flow problem. *J Assoc Comput Mach*, 35:931–940, 1988.

[18] E. Gourdin, M. Labbé, and H. Yaman. *Facility Location: Applications and Theory*, chapter Telecommunication and Location, pages 275–305. Springer, 2001.

[19] D. Griffith and S. K. Lee. A 1+1 protection architecture for optical burst switched networks. *IEEE Journal on Selected Areas in Communications*, 21(9):1384–1398, November 2003.

[20] M. Grötschel, C. L. Monma, and M. Stoer. Design of survivable networks. In M. B. et al., editor, *Handbooks in OR & MS*, volume 7, pages 617–671. 1995.

[21] M. Grötschel, C. L. Monma, and M. Stoer. *Network Models*, chapter Design of Survivable Network Design, pages 617–671. Elsevier Science, 1995.

[22] J. Hao and J. B. Orlin. A faster algorithm for finding the minimum cut in a directed graph. *J Algorithms*, 17:424–446, 1994.

[23] C. Huang, V. Sharma, K. Owens, and S. Makam. Building reliable mpls networks using a path protection mechanism. *IEEE Communications Magazine*, 40(3):156–162, March 2002.

[24] H. Kerivin and A. R. Mahjoub. Separation of partition inequalities for the (1,2)-survivable network design problem. *Operations Research Letters*, 30:265–268, 2002.

[25] H. Kerivin and A. R. Mahjoub. Design of survivable networks. *Networks*, 39:81–87, February 2005.

[26] H. Kerivin and A. R. Mahjoub. On survivable network polyhedra. *Discrete Mathematics*, 290:183–210, 2005.

[27] H. Kerivin, A. R. Mahjoub, and C. Nocq. *The Sharpest-Cut*, chapter (1,2)-Survivable Networks: Facets and Branch-and-Cut, pages 121–152. MPS/SIAM Optimization, 2004.

[28] S. W. Kim and S. W. Seo. Regenerator placement algorithms for connection establishment in all-optical networks. *IEE Proceedings Commun.*, 148(1):25–30, February 2001.

[29] J. G. Klincewicz. Hub location in backbone/tributary network design: a review. *Location Science*, 6:307–335, 1998.

[30] M. Labbé, G. Laporte, I. R. Martin, and J. J. S. Gonzalez. The ring star problem: Polyhedral analysis and exact algorithm. *Networks*, 43(3):177–189, 2004.

[31] M. Labbé and H. Yaman. Solving the hub location problem in a star-star network. *Networks*, 51:19–33, 2008.

[32] M. Labbé, H. Yaman, and E. Gourdin. A branch and cut algorithm for hub location problems with single assignment. *Mathematical Programming*, 102:371–405, 2005.

[33] C. Li, S. T. McCormick, and D. Simchi-Levi. The complexity of finding two disjoint paths with min-max objective function. *Discrete Applied Mathematics*, 26:105–115, 1990.

[34] A. R. Mahjoub. Two-edge connected spanning subgraphs and polyhedra. *Mathematical Programming*, 64:199–208, 1994.

[35] A. R. Mahjoub. On perfectly two-edge connected graphs. *Discrete Mathematics*, 170:153–172, 1997.

[36] B. Mukherjee. Wdm optical communication networks:progress and challenges. *IEEE Journal on Selected Areas in Communications*, 2000.

[37] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. Wiley-Interscience, 1999.

[38] G. Reinelt. Tsplib - a traveling salesman problem library. *ORSA J Comp*, 3:376–384, 1991.

[39] G. Shen and W. D. Grover. Segment-based approaches to survivable translucent network design under vaious ultra-long-haul system reach capabilities. *Journal of Optical Networking*, 3(1):1–24, January 2004.

[40] M. Stoer. *Lecture Notes in Mathematics-Design of Survivable Networks*. Springer-Verlag, 1992.

[41] J. Strand, A. L. Chiu, and R. Tkach. Issues for routing in the optical layer. *IEEE Communications Magazine*, 39:81–87, February 2001.

[42] D. Vandenbussche and G. L. Nemhauser. The 2-edge-connected subgraph polyhedron. *Journal of Combinatioral Optimization*, 9:357–379, 2005.

[43] J. Vygen. Np-completeness of some edge-disjoint path problems. *Discrete Applied Mathematics*, 61:83–90, 1995.

[44] L. A. Wolsey. *Integer Programming.* Wiley-Interscience, 1998.

[45] H. Yaman. Polyhedral analysis for the uncapacitated hub location problem with modular arc capacities. *SIAM Journal on Discrete Mathematics*, 19(2):85–111, 2005.

[46] X. Yang and B. Ramamurthy. Dynamic routing in translucent wdm optical networks. In *Proceedings of IEEE ICC'2002*, New York,NY, 2002.

[47] X. Yang and B. Ramamurthy. Sparse regeneration in translucent wavelength-routed optical networks: Architecture, network design and wavelength routing. *Photonic Network Communications*, 10(1):39–53, 2005.

[48] Y. Ye, T. H. Cheng, and C. Lu. Routing and wavelength assignment algorithms for translucent optical networks. *Optics Communications*, 229:233–239, 2004.

[49] E. Yetginer and E. Karaşan. Regenerator placement and traffic engineering with restoration in gmpls networks. *Photonic Network Communications*, 6(2):139–149, 2003.