Models and Algorithms for Telecommunication Network Design

Promotor:

Prof. dr. ir. A.W.J. Kolen

Co-promotor:

Dr. ir. C.P.M. van Hoesel

Beoordelingscommissie:

Prof. dr. H.J.M. Peters (voorzitter)
Prof. dr. S.H. Tijs
Prof. dr. L.A. Wolsey (Universite Catholique de Louvain)

Models and Algorithms for Telecommunication Network Design

Proefschrift Universiteit Maastricht
ISBN 90-9012781-X

Acknowledgements

Apart from it being a challenge, the most important reason for starting a PhD project was a large list of topics in the field of Operations Research that interested me, but were beyond the scope of my knowledge at that time. After four years of research I am happy to conclude that I was able to stand up to the challenge and become acquainted with many of these issues. Fortunately, each answer was replaced by several new challenges, although I must admit that the corresponding subjects were far from limited to the scientific field. Many people have contributed to this process and many more should receive credit for their friendship and support throughout the years.

First of all, I owe a lot to Antoon Kolen. His enthusiasm and desire to solve new scientific problems that cross his path have often been an inspiring factor. He provided an open atmosphere that allowed me to explore new areas and undertake several enjoyable trips to foreign conferences. Stan van Hoesel has been equally important. His door was always open, even if he was not there, and I have greatly benefited from his wide knowledge. Sincere apologies are in place for being the stubborn PhD student who on many occasion thought he new better. Olaf Flippo left Maastricht University during my PhD project but I am grateful for the time I could spend around such a sincere and happy person. Joris van de Klundert gave me confidence in my work and was always willing to share his musical interests with me.

For almost four years, Arie Koster served as my roommate, on-line computer help, co-traveler and navigator on foreign trips, discussion partner, telephone operator, co-researcher on several subjects, and above all, as a friend. I have greatly enjoyed our open discussions on a large variety of subjects, even though we may have had opposite opinions at times. Thanks to Jos, Rudolf, Jan-Willem, Karin, Yolanda and the other people from the Department of Quantitative Economics. Special gratitude to Hans van Kranenburg for being a pleasant neighbor both at the office and at home, our games of squash, early morning swimming activities, and for his inspiring views on things such as soccer and politics. Thanks also to the Econometrics students that had the custom of visiting room 4084 at regular time intervals. In particular, Anton v.d. Kraaij proved to be the perfect partner during a joint computer programming project.

I am also indebted to many people outside Maastricht University. Frank van der Duyn Schouten was the first to introduce me to the field of Operations Research, and he should also be thanked for pointing me to research opportunities in the academic society. Stef Tijs, Peter Borm, Herbert Hamers and Vincent Feltkamp were cooperative players during

my game theoretical period prior to this PhD thesis. Karen Aardal repeatedly showed interest in my progression and invited me to some conferences. Martin Savelsbergh, Daniel Bienstock, David Williamson, Oktay Günlück, and Raghavan were perfect hosts during a research trip in the United States. Shell provided financial support that allowed me to visit an INFORMS conference in Israel, and Bram Verweij did some magnificent photography work on that trip. The people at KPN Research, Leidschendam are thanked for their patience in explaining technical telecommunication details and their interest in the research progress. This holds especially for Hans van de Berg, Cor Lavrijsen, Jaap Geerdink, Saskia Vlaar, Bart Klein Obbink, Maurits de Graaf and Jeroen Warmerdam.

Thanks to the players of indoor soccer team Deo Volente in Tilburg for the many taxi rides in between the central station and some distant sports complex, free accommodation after late night parties, as well as an immeasurable amount of fun during the actual soccer matches. I am grateful to Marco Verstappen for his technical support in motorcycle maintenance and for lending me his red Toyota Starlet during a cold Dutch winter that happened to coincide with a hot Surinam summer. My friends in Helenaveen are greatly appreciated for being exactly that.

Constant support and love was given by Wendy Beek. Apologies for the numerous times when I was more interested in mathematical formulas and late night computer programming than in socializing. Hopefully the near future will be a good time to catch up. Special thanks also to Henk and Marja Beek for their unlimited hospitality.

Mijn grootste dank gaat uit naar mijn ouders, Wim en Gerarda van de Leensel, zonder wie dit alles niet mogelijk zou zijn geweest.

May 1999
Robert van de Leensel

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction

Telecommunication networks are part of a dynamic environment. Ever since the invention of telephony the telecommunication markets have registered a continuous growth in the demand for telecommunication services. This trend is expected to pursue in the near future. Moreover, rapid progress in telecommunication technology has had an enormous impact on the design and scope of networks. On the one hand, it has enabled instant worldwide communication by means of modern communication devices and has triggered the introduction of other than traditional voice transmission services. Nowadays telephone, computer and data networks are widely present in practically all sectors of the (inter)national economy. On the other hand these innovations have reduced the costs of providing telecommunication services, thereby improving its profitability. In addition, the global deregulation of the telecommunication industry has made the market more competitive, forcing telecommunication companies to operate more efficiently.

Telecommunication networks are continuously adapted in response to these changing environments. During this process numerous decisions have to be made which influence a telecommunication network's performance characteristics such as its speed, capacity, security, availability, reliability, maintainability, and not at the least, its costs. Each year the telecommunication industry invests billions of dollars worldwide to maintain and upgrade their networks. As a result, significant cost savings opportunities arise in the design of telecommunication networks. The issues that have to be addressed in order to achieve cost reductions are usually very complex and highly interdependent. Techniques from the scientific field of combinatorial optimization have proven to be a successful approach in the modeling and analysis of a variety of telecommunication network design problems.

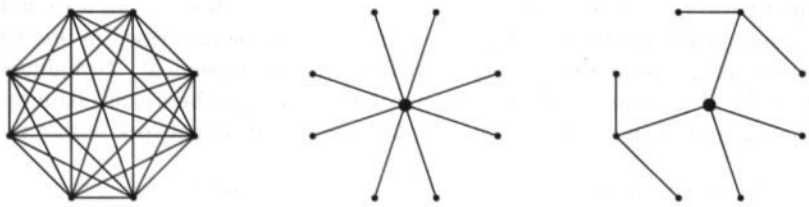
The aim of the current study is to develop guiding procedures for recent topics in the telecommunication network design process. This dissertation therefore reports on models and algorithms for a number of problems in the topological design of telecommunication networks, in which cost-efficient telecommunication networks satisfying certain performance criteria are to be designed. As an important byproduct, new theory in the field of

combinatorial optimization is developed which may lead to a better understanding and an improved capability of solving related problems in the future.

In this introductory chapter we give a detailed outline of the telecommunication network design problems studied in this thesis. Since current and future network design issues are significantly affected by previously made decisions and therefore the state of the telecommunication network as it is, Section 1.2 presents a brief overview of the topology of telecommunication networks, focusing on the topics that are relevant for the problems in this thesis. At the same time this will lead to a deeper understanding of the variety of problems arising in the field of telecommunications as well as the commonly accepted terminology. Section 1.3 treats the development in telecommunication technology in recent years which forms the basis of the majority of problems studied in this thesis. The background and settings of the specific network design issues considered in our study are addressed in Section 1.4. Finally, Section 1.5 functions as a guide through the subsequent chapters of this thesis.

1.2 The Topology of Telecommunication Networks

The first telephone networks connected a number of customers in a small geographical area via a mesh-network (see Figure 1.1(a)). Although fully operational, such networks can indeed be cumbersome and inconvenient because they require a direct link between every pair of customers. A significant reduction in the number or required links can be obtained by using a star topology as illustrated in Figure 1.1(b). Here, all customers are connected to a *switching* center, which has the ability to establish a connection between two customer lines and thereby enables communication between the corresponding customers. Additional link savings can even be implemented by means of a tree network as depicted in Figure 1.1(c). Customers are not necessarily connected directly to the switching center, but instead, such a connection may pass several other customers. However, even though customers might be geographically close and interconnected via a direct link, all switching functionality, i.e. the ability to establish communication between customers in the network, is still performed by the switching center. Tree networks are currently the most widely used topology to interconnect a number of customers in a small geographical area.



(a) mesh topology (b) star topology (c) tree topology

Figure 1.1: Example Network Topologies

To enable communication between customers located in different geographical areas (the different tree networks), these local networks have been grouped into regional networks, which in turn are interconnected into national networks, until finally nationwide communication between any pair of customers can be established. Although the exact number of layers may vary from country to country, most existing telecommunication networks are based on a hierarchical structure, and international networks are designed accordingly. Consequently, the size of the geographical region and the amount of telecommunication traffic handled by an individual switch or link in the network decreases as one moves down to the lower layers in the hierarchy. Next, we discuss this hierarchical network topology in more detail.

Consider the four-level network depicted in Figure 1.2. The top level of the network is usually referred to as the *backbone network*. The high capacity switches in the backbone serve large regions of a country (typically multiple provinces or states), and these switches are interconnected via high speed, high capacity links. The overall connectivity (number of links between the switches) in the backbone is usually high. On the one hand, large traffic flows which emerge in this part of the network make it economically attractive to have direct connections between different regions of a country. On the other hand, as each switch or link serves a large portion of the total telecommunication service provided by the network, a single backbone network component failure (breakdown of a node or a link) could cause a significant reduction on the network's performance. By increasing the connectivity in the backbone network, alternative routing schemes can be employed for traffic flows affected by a network failure.

In the *switching network*, lower capacity switches are installed which serve (parts of) individual states, provinces or cities and a typical topology used here is a ring structure, providing two possible access routes for each switch on this level to the backbone network. Several of these rings may be connected to a single backbone switch, and the switching network itself could even consist of multiple hierarchical layers. The switching network is however the lowest layer in the network hierarchy which can perform the switching functionality.

Moving to lower layers in the hierarchical structure, the number of customers served by a node in the network and the amount of telecommunication traffic transmitted on a link decrease. From an economical point of view it is therefore less interesting to have a high connectivity between nodes in the lower layers of the network. Moreover, a network failure of a low level network component would only have an impact on a small group of customers. The topology one often encounters in lower layers of the network hierarchy is therefore a tree structure, as depicted in the third layer of Figure 1.2. This *feeder network* consists of a number of distribution points (typically ranging from a few dozens to a few hundreds) that are connected via a unique path to a switch in the switching network. Each of these distribution points serves a collection of customers (possibly a few hundreds) in an underlying street network which is called the *distribution network*. The combined distribution and feeder network are usually referred to as the *Local Access Telecommunication Network*, because they provide the access of customers to the switching functionality of the higher layers in the hierarchy.

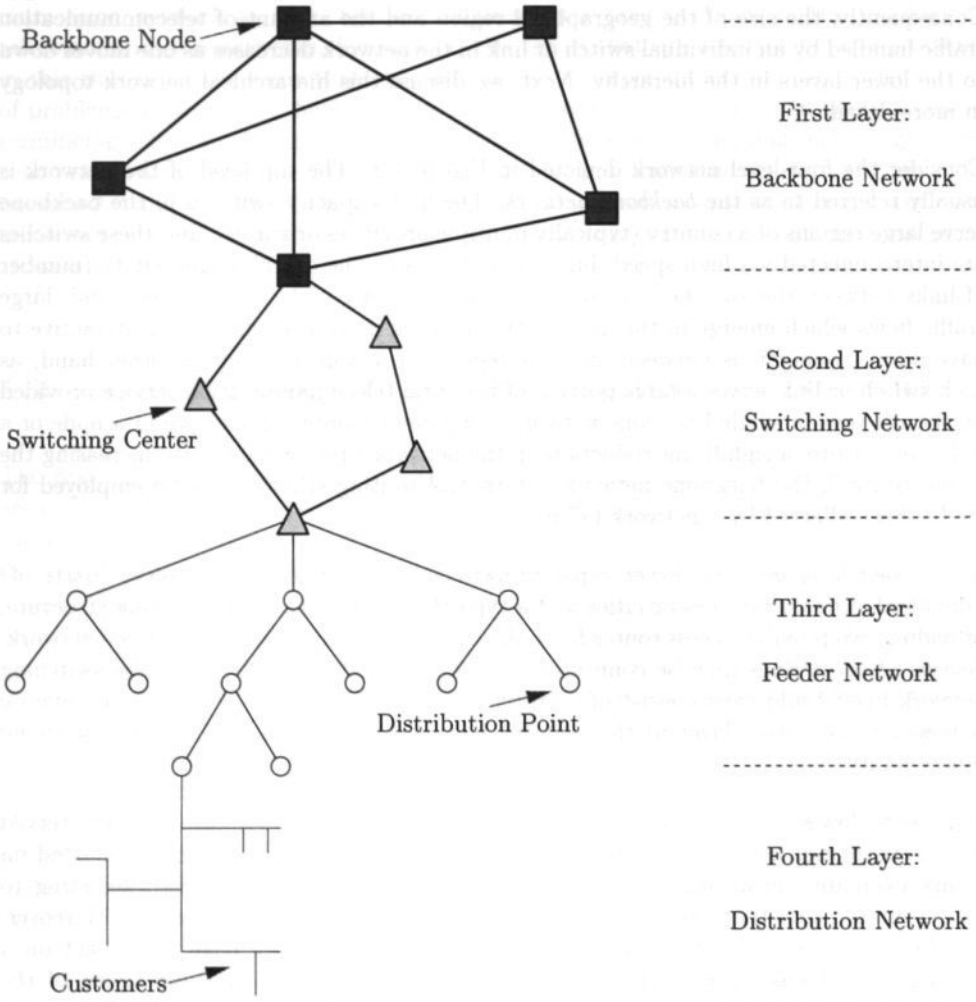


Figure 1.2: Topological Network Hierarchy

The above describes an operational hierarchical network structure. In practice, many components of the network might be subject to breakdown, due to deterioration, maintenance, climatological influences, or human mistakes. Telecommunication companies often find it desirable to take additional measures in order to prevent a decline of telecommunication service in case of a network failure. This can be achieved by the installation of so-called backup components in the network, which may function as surrogates of components that are subject to failure. In strategically important parts of the network (such as the backbone network) dedicated backup networks are even encountered in practice.

In the topological design of telecommunication networks as described in the above, numerous decisions have to be made. These include the following:

- How many layers should the network hierarchy contain?
- How should the different layers of the hierarchy be interconnected?
- How many switches (or other telecommunication devices) should each layer of the network contain, and where should they be located?
- What network topology (star, tree, mesh, arbitrary) should be used in each layer of the network hierarchy, that is, how should the switches in the same layer be interconnected?
- How much capacity should be installed on all the links in the network to accommodate all traffic flows?
- What type of routing schemes should be employed to deliver traffic flows from origin to destination?
- How should network failures be handled?
- How much additional capacity must be installed in order to cope with breakdowns?

All of these issues (and many others) are part of the *topological network design process.* Obviously, telecommunication companies would like to build the optimal network topology, but one could evaluate different network topologies by means of several criteria:

- the average delay between admission of the message to the network and the delivery at the destination, and/or the variability in this delay,
- the capacity of the network measured as the maximal amount of telecommunication traffic per time unit the network can transmit,
- the availability of service and the probability of information loss during transmission,
- the reliability of the network in case of network component failures,
- the network's information security, i.e. the level of privacy of transmitted data,
- the expected current and discounted future costs of the network.

Of course a combination of these criteria could also be exploited. Unfortunately, some of these criteria are contradictory. For instance, a high quality of service (small delay, high reliability) would definitely be achieved by mesh networks containing large capacity network components, but this would be in conflict with economical guidelines which emphasize the importance of efficient use of capacity. In this thesis we will adopt the economic approach: the goal is to build cost-efficient network topologies that satisfy certain network performance characteristics. Moreover, we will focus primarily on four parts of the overall topological decision process, namely the location of telecommunication devices in networks, the routing of telecommunication messages through networks, the capacity installation on links of the network, and the implementation of spare capacity to handle network component failures.

1.3 Integrated Services Digital Network (ISDN) and the Asynchronous Transfer Mode (ATM)

Although the discussion so far has been limited to telephone networks, a similar evolution and the same type of problems arise in all telecommunication networks (computer networks, data networks) or any transportation network in general. Traditionally, telecommunication companies implemented separate networks for different telecommunication applications such as voice and data transmission. Since each individual network requires dedicated equipment, interfaces and protocols in order to provide the telecommunication service to customers, the adherence of such a strategy often leads to costly overall solutions requiring extensive management. In 1984 the ITU (International Telecommunication Union) therefore formalized the idea of a single network capable of providing multiple services ([49]):

> "... an ISDN (Integrated Services Digital Network) is a network ... that provides end-to-end digital connectivity to support a wide range of services, including voice and non-voice services, to which users have access by a limited set of standard multi-purpose user-network interfaces."

Ever since, ISDN has been developed to a completely digital network, with a single transfer mechanism to serve distinct applications such as voice data and image transmission. Digital information transfer has several advantages over analog information transfer. Firstly, digital information can be transferred over large distances without the loss of information. Secondly, digital information can be stored and processed directly by computers, which enables efficient communication between computers. Thirdly, errors in digital information can easily be detected and corrected. Finally, digital information transfer can be achieved at lower costs than its analog variant. Hence, in contrast to several special-purpose networks, ISDN-networks can offer significant economic advantages such as in development, implementation, operation and maintenance.

The development of ISDN has been greatly affected by technological innovations such as the introduction of fibre optic transmission, which enables the high speed, high ca-

pacity transmission of digital data between remote locations. Moreover, fibre cables have proven to be highly reliable, and easy to install. Inspired by this technological progress in transmission and switching techniques, many new telecommunication services have been introduced, including tele-shopping and tele-advertising, remote education and training, database retrieval, entertainment applications such as video-on-demand and remote games, video surveillance, video-conferencing and electronic publishing. Since many of these services require high transmission rates in order to achieve high quality image applications, the concept of Broadband-ISDN (B-ISDN) has been introduced which enables such high capacity telecommunication traffic streams. One of the key ideas of B-ISDN is its flexibility in capacity assignment which would enable current and future services to be incorporated into the same, single network.

Obviously, several of these services have different characteristics. For instance, traditional voice transmission requires a small but constant capacity on the connection between sender and receiver of the message. Moreover, the delay and variance in the delay of the message need to be small in order to guarantee a high quality of the speech transmission, whereas a small, marginal loss of information will be imperceptible to users. In contrast, data transmission may require a variable capacity connection, its delay may be less important, but any loss of information will probably be unacceptable. The integration of such distinct services on a single network therefore asks for strict agreements on matters as capacity assignment, traffic priorities, delay, and loss of information. Although still under elaboration, ATM (Asynchronous Transfer Mode) is currently accepted as the transfer mode to perform the transmission and switching functionalities of B-ISDN networks.

In the Asynchronous Transfer Mode the information of a telecommunication message, which is sent from an origin to a destination, is stored in a number of cells. Each of these ATM-cells consists of two parts, namely a header field and an information field (cf. Figure 1.3). The information field contains a small part of the actual telecommunication message. The header field contains all information required to route the cells from origin to destination in such a way that the service is provided at its desired level of performance. Moreover, it contains the necessary data required to reconstruct the original telecommunication message from the separate data pieces at the point of destination.

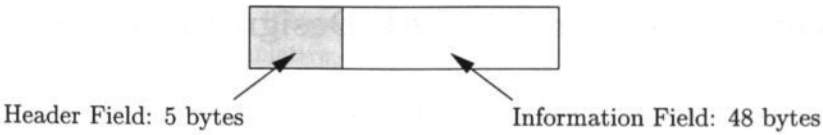
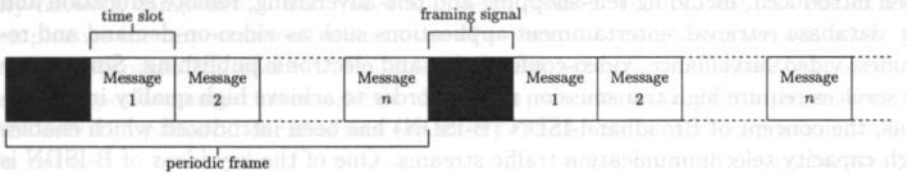


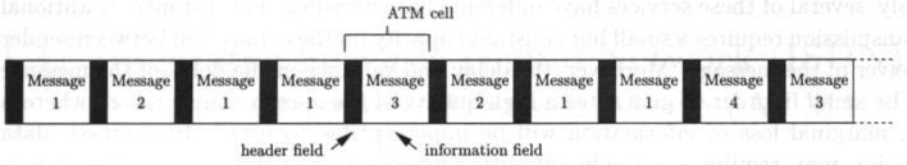
Figure 1.3: ATM Cell Layout

In traditional synchronous transfer modes (see Figure 1.4(a)) information from telecommunication messages is transmitted through the network by an identical capacity assignment for each message in each periodic frame. Consequently, both the time interval between two consecutive information parts from a telecommunication message, as well as the amount of information routed through the network for a message are constant for each telecommunication message. In contrast, the *asynchronous* transfer mode allows for arbitrary assignments of cells to telecommunication messages in the stream of information transferred through the network, leading to irregular patterns as depicted in Figure 1.4(b).

As such, the asynchronous transfer mode allows for the integration of telecommunication services with different capacity and service requirements into a single network.



(a) Synchronous Mode



(b) Asynchronous Mode

Figure 1.4: Transfer Modes

Eventually, B-ISDN networks are expected to replace existing telecommunication networks, although a lot of the existing infrastructure in current networks can be (re)used in the design of ATM-based B-ISDN networks. Again, many similar topological network design problems as mentioned in the above arise in the development of ATM-networks. Decisions regarding the location and inter-connectivity of telecommunication devices have to be made. In the subsequent section we will discuss some of these problems as discussed in this thesis in more detail.

1.4 Telecommunication Network Design Problems

Ideally, one would like to solve the overall topological network design problem as a whole, because the design issues in the distinct layers of the hierarchical structure are heavily interdependent. Since this problem is by far too complex to analyze given the current state of computer technology and human knowledge on these problems, they are often decomposed into smaller problems. Traditionally, network design problems have been separately analyzed for each layer of the hierarchy, and even within the same layer design issues such as the location of telecommunication devices, capacity installation on links, and the routing strategies of traffic flows have been studied independently for reasons of problem complexity. Gavish [38],[39] schematizes the overall network design problem and its decomposition techniques, and states useful references to literature on the individual problems. For a more recent taxonomy on network design problems as well as an extensive list of references we refer to Chapters 18 and 19 of the Annotated Bibliographies in

Combinatorial Optimization [54]. In the subsequent subsections we discuss two areas of network design which have received a lot of attention in literature and which form the basis for the problems studied in this thesis.

1.4.1 Topological Design Problems on Tree Networks

An important part of the telecommunication networks existing today is formed by the Local Access Telecommunication Network, i.e. the part of the network between the customers and the lowest level switches in the network (as discussed in Section 1.2). This access network can again be viewed as a hierarchical structure, as depicted in Figure 1.5. A switching center is located in the root of the tree which serves a number of distribution points (typically in the range of a few dozens to a few hundreds). Each of these distribution points serves an underlying distribution network, consisting of a number of individual customers (cf. Figure 1.2). Because the only switching functionality is located in the root of the tree, all communication with customers within the access network is handled via the switching center in the root of the access network. Moreover, since this access network has a tree structure there only exists a single path from each distribution point to the switching center, hence, the routing of these telecommunication messages through the access network is easy.

Figure 1.5 shows an access network which displays both the estimated future demand for telecommunication services for each distribution point. This estimated future demand represents the number of (simultaneous) connections required between the distribution point and the switching center to obtain the desired level of service. Moreover, the figure displays the available capacity on each link in the network, in the same unit of measurement as the demand figures. Because of the unique path from a distribution point to the switching center, the required capacity on a link is equal to the sum of demands of all distribution points in the subtree located underneath the link. Links whose capacity cannot accommodate the total future demand of this subtree are depicted as an emphasized line. For instance, in Figure 1.5, the link between node 2 and node 1 must be able to route the demand of node 2 to the switching center, hence, this link has sufficient capacity. However, the link between node 1 and the switching center must accommodate the cumulative demand of the nodes $1, 2$ and 3 which equals 120. Since the current capacity on the link equals 90 the link has insufficient capacity to enable the estimated future communication.

Traditionally, if an increase in demand for telecommunication services resulted in links with insufficient capacity, additional capacity installation on the links was the only solution to satisfy future communication. The introduction of multiplexers and concentrators changed this situation. Concentrators and multiplexers are electronic devices which can compress signals from several incoming connections into a combined outgoing signal which can be transmitted on a single or relatively few high frequency connection(s) (although there are some technical differences between concentrators and multiplexers, for the purpose of our problem description their behaviour is alike, and we will refer to them as concentrators henceforth). As a consequence, the installation of a concentrator in the network can reduce the amount of required capacity on the links in the network. Some

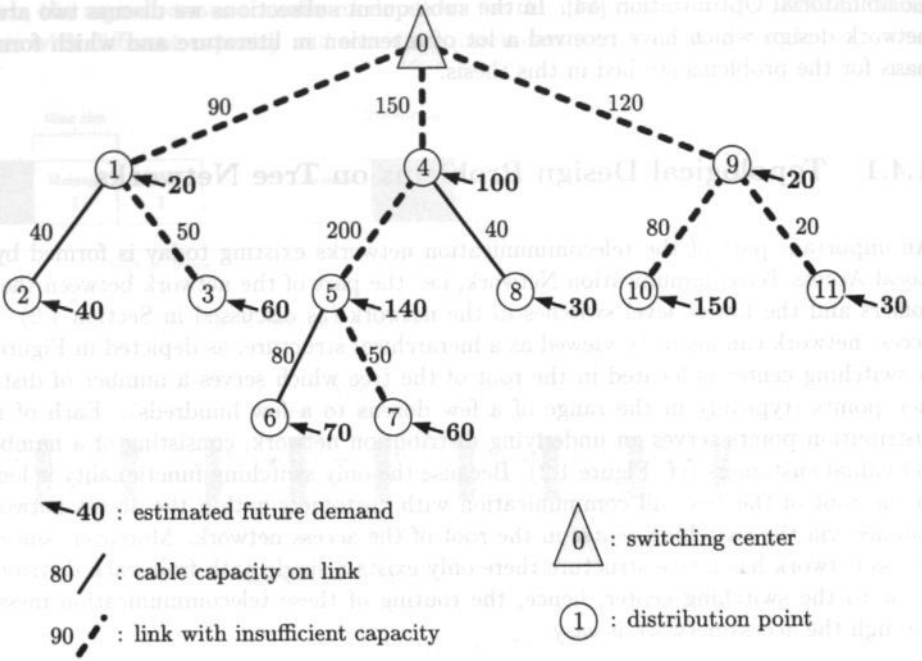


Figure 1.5: Local Access Telecommunication Network with Insufficient Capacity

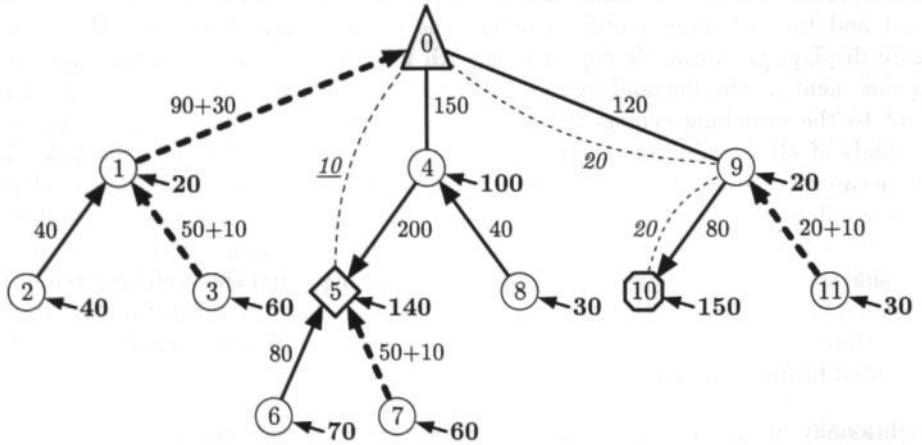


Figure 1.6: Local Access Telecommunication Network with Sufficient Capacity

of these concentrators may use the same type of connection as already available in the network to route compressed traffic to the switching center, whereas other concentrators require dedicated high capacity point-to-point connections between the concentrator and the switching center.

Reconsider Figure 1.5, then the following measures could be taken in order to enable all estimated future demand, as illustrated in Figure 1.6. Capacity expansion on links $(3,1)$ and $(1,0)$ guarantees sufficient capacity to accommodate the demands of nodes $1,2$ and 3. Capacity expansion can also be applied on link $(7,5)$ to accommodate all demand of node 7. Furthermore, a concentrator is installed in node 5. This concentrator serves the nodes 4 through 8, with a cumulative demand of 400. If the concentrator requires a point-to-point connection with the switching center and has a compression ratio $40:1$ (the ratio between number of incoming connections and required number of outgoing connections), then the number of outgoing connections on this point-to-point connection equals 10. Finally, we expand the capacity on link $(11,9)$ install a concentrator with compression ratio $10:1$ in node 10 to serve nodes 9 through 11, with a cumulative demand of 200. The compressed signals are routed from the concentrator to the switching center using the same connections as already available in the network.

The compressing capability of concentrators yields a reduction in the required capacity on the links of the network. Such a reduction can also be achieved by installing *remote* switches in the network. Customers corresponding to distribution points that are connected to this remote switch can communicate with each other without the interference of the switching center located in the root of the tree. For instance, suppose that 40 connections of the estimated future demands of nodes $1,2$ and 3 corresponds to mutual communication. If in Figure 1.5 a remote switch is installed in node 1, then all communication between nodes $1,2$ and 3 can be performed by the remote switch in node 1. As a result, only the remainder of the total demand of nodes $1,2$ and 3 (equal to 80) needs to be routed to the switching center, and therefore the link $(1,0)$ needs no capacity expansion. Hence, the installation of a remote switch also reduces the required capacity in other parts of the access network. Moreover, apart from the switching functionality a switch can usually also perform traffic compression as performed by a concentrator.

Summarizing, an increase in the demand for telecommunication services can now be treated either by capacity expansion on the links of the network, the installation of concentrators and/or remote switches, ór by a combination of these measures. Since different solutions are accompanied by different costs, a topological network design problem arises in Local Access Telecommunication Networks: the objective is to find the cheapest solution for the expansion of the capacity of the access network such that all estimated future demand can be handled by the access network.

1.4.2 Topological Design Problems on General Networks

The second type of problems we discuss in this thesis arises in higher layers of the telecommunication network hierarchy. Here, the decision concerning the location of the telecommunication devices has often already been made. However, in contrast to tree networks,

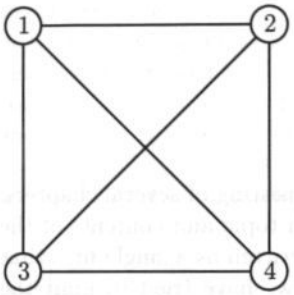
the connectivity in these layers is usually higher, and as a consequence multiple routing strategies can be exploited for telecommunication traffic flows. Moreover, available telecommunication technologies often imply that the quantity of installed capacity on links of the network is restricted to a discrete set of capacity amounts. Hence, spare capacity may arise in the networks, which can be exploited to route additional traffic flows on the link without incurring higher capacity costs. Different routing strategies can therefore correspond to different capacity installation requirements on the links of the network, and consequently, different network costs. Network loading problems therefore aim to solve the combined problem of designing routing strategies for telecommunication traffic flows and capacity installation on the links of the network.

Consider for instance the four node network depicted in Figure 1.7(a), where each node represents a switch in the backbone network. The lines in the picture represent the links that can be used for routing of telecommunication traffic between the backbone switches and we assume that all possible links are available. The estimated demand for telecommunication services between the nodes in the network is given by Figure 1.7(b). For example, we have to send 13 units on a single path from node 2 to node 4, and vice versa. In order to enable the transmission of telecommunication traffic flows between the switches, sufficient capacity has to be installed on the links of the network. Suppose that capacity on links can only be installed in multiples of ten, and the required capacity on a link must be greater than or equal to the sum of traffic flows in both directions on the link. Next we consider some different network configurations.

If all traffic flows are routed directly from their origin to destination (without any intermediate nodes) then capacity installation is needed on each link in the network. Consider the link between node 2 and node 4. The total flow on the link is $13 + 13 = 26$, and since capacity can only be installed in multiples of ten, three capacity units must be installed on this specific link. The same analysis can be applied to the remaining links of the network and the resulting capacity installations are depicted in Figure 1.7(c). Consequently, this routing strategy requires the installation of 20 capacity units in the network.

Next, we consider a different routing strategy. First, suppose that all telecommunication demands with size greater than or equal to ten are routed directly from origin to destination. This routing strategy is shown in Figure 1.7(d), where the numbers represent the resulting flow on the links. Since capacity can only be installed in multiples of ten, on all of these links we could route some additional traffic without incurring higher costs. For example, the traffic flow on link $(1, 4)$ currently equals 62 which would require 7 capacity units. Hence, this link can accommodate an additional traffic flow of 8 without incurring higher costs. This idea may lead to the routing strategy for the remaining traffic flows as represented in Figure 1.7(e). The combined routing strategy of Figures 1.7(d) and 1.7(e) thus yields a total capacity requirement of 18 capacity units, as depicted in Figure 1.7(f). This shows that different routing strategies may correspond to different network costs.

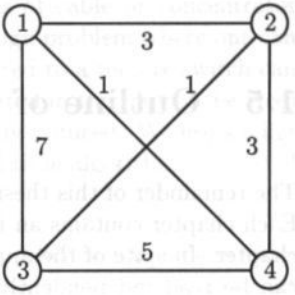
In addition, telecommunication companies often want to protect their networks against component failure such as the breakdown of a switch or a link. Several measures can be taken to improve the reliability of a network. Spare capacity could be installed in the network which can be used to obtain multiple routing opportunities for traffic flows.



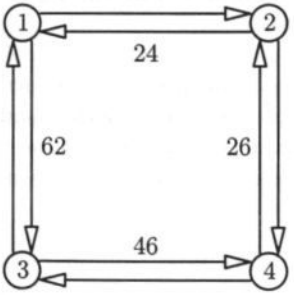
(a) Network

	1	2	3	4
1	0	12	31	3
2	12	0	1	13
3	31	1	0	23
4	3	13	23	0

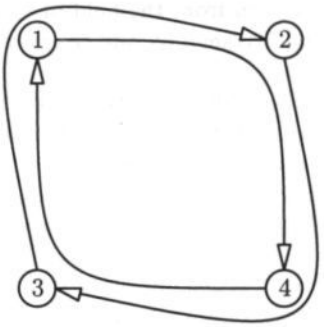
(b) Demands



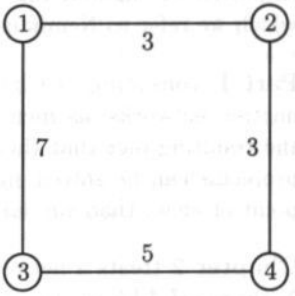
(c) Solution



(d) Partial Routing



(e) Partial Routing



(f) Solution

Figure 1.7: Example Network Loading Problem

Alternatively, separate backup networks could be employed to minimize the consequences of breakdowns. Instead of designing reliable networks capable of coping with network failures, one could also suggest routing strategies which limit the amount of traffic flow through a node or link of the network. That way, the amount of telecommunication traffic affected by a node or link breakdown is limited accordingly.

Summarizing, given a network, estimated future demand between nodes of the network, and capacity installation costs, a topological network design problem arises if the estimated demand exceeds current capacity in the network. The objective is to simultaneously design routing strategies and capacity expansion schemes such that costs are minimized and certain reliability criteria are satisfied.

1.5 Outline of the Thesis

The remainder of this thesis is divided into two parts, each consisting of several chapters. Each chapter contains an introduction on the specific research topic and contents of the chapter. In spite of the apparent connection, each chapter is written as a single unit that can be read independently of the other chapters. Although we have tried to limit the involved repetition of definitions and models to an absolute minimum, we apologize to those who please the author by reading multiple chapters. The mathematical techniques employed throughout this thesis stem from the field of Combinatorial Optimization, for which we refer to Nemhauser and Wolsey [62] and Schrijver [68].

Part I, consisting of Chapters 2, 3 and 4, treats topological design problems defined on tree networks, as mentioned in Section 1.4.1. By exploiting the tree structure and the resulting fact that there is a unique path between any pair of nodes, many of these problems can be solved significantly faster (both from a theoretical as from a practical point of view) than similar problems defined on arbitrary graphs.

Chapter 2 treats a network design problem arising in Local Access Telecommunication Networks (LATN's). In a LATN the only switching functionality is available by means of the switching center located in the root of the tree and hence all communication messages pass through the switching center in the root. As demand for telecommunication services grows, the available capacity in the network might not be sufficient to accommodate all traffic flows. Given the estimated future demand for telecommunication services and the current transmission and compression capacity, additional capacity should be installed in the network such that all traffic demand can be transmitted to the switching center. In the *Local Access Telecommunication Network Expansion Problem* (LATNEP) capacity in the network can be expanded by means of capacity expansion on the links, as well as by concentrator installation. The key issue in LATNEP is therefore to find an efficient trade-off between cable expansion and concentrator installation costs. Many problems related to capacity expansion in LATN's have been studied in the literature (see Gavish [38] and Balakrishnan, Magnanti and Wong [8] for an overview). For some time, the exact complexity of LATNEP was unknown. Balakrishnan, Magnanti and Wong [9] state that the problem is NP-hard and use a Lagrangean relaxation incorporating valid inequalities to

obtain solutions for real-life instances within 7% of optimality. Cho and Shaw [22] apply a limited column generation technique on randomly generated instances of LATNEP. Flippo, Kolen, Koster and van de Leensel [33] prove that LATNEP is actually *weakly* NP-hard and can be solved by a pseudo-polynomial time algorithm. Chapter 2 reports on this algorithm and shows how problem instances previously studied in the literature (as well as significantly larger instances) can be solved efficiently.

Chapter 3 presents the research on the design of an ATM tree overlay network problem performed in cooperation with KPN Research, Leidschendam, the Netherlands. The idea is to install ATM switching and transmission functionality on an existing tree network in such a way that all future demand can be accommodated by the network. In contrast to chapter 2 where capacity can be expanded only by means of cable or concentrator installation, Chapter 3 discusses a topological tree network design problem where one can also install remote switches in the network. Customers connected to a remote switch can communicate via this nearby switch (instead of the switch located in the root of the tree) and thereby the overall traffic in the tree network is significantly reduced. We use similar ideas as exploited for LATNEP to obtain a psuedo-polynomial time algorithm capable of solving real-life data made available by KPN Research.

Many topological tree network design problems cannot be solved efficiently using a psuedo-polynomial time dynamic programming algorithm, consequently one is forced to resort to other solution methods. In **Chapter 4** we therefore investigate the polyhedral structure of the so-called Precedence Constrained Knapsack Problem (PCKP) which arises frequently as a substructure in mathematical models for topological design problems on tree networks. This substructure consists of a knapsack constraint and a set of precedence constraints. The knapsack constraint usually represents the limited capacity of a telecommunication device (switch or concentrator) located in the tree network. Due to operational restrictions a node in the tree network can often only be served by such a device if all nodes on the unique path from the node to the device are served by that device. Such restrictions are modeled by precedence constraints. We study classes of valid inequalities, the complexity of lifting valid inequalities and the improvements these inequalities can yield in a computational study.

Part II, consisting of Chapters 5 and 6 deals with combined routing and capacity installation problems on general network structures. The common basis for these problems is as follows. One is given an arbitrary network and a number of traffic demands. Each of these demands must be routed from its origin to its destination through the network. In order to accommodate the resulting traffic flows sufficient capacity must be installed on the links in the network. The goal of these so-called network loading problems is to simultaneously design routing schemes and capacity installation schemes which minimize the overall routing and capacity installation costs. In practice, many problem variants arise, caused by differences in routing restrictions, available capacity types, reliability conditions and cost functions.

Chapter 5 introduces a set of network loading problems, the subject of study in a joint research project with KPN Research, Leidschendam, the Netherlands. We mention the underlying ideas of these models and the motivation to study these problems. Given the

multiplicity, diversity and complexity of these problems, developing fast exact solution methods for this complete set of network loading problems is expected to be impossible. Therefore, we report on local search heuristics implemented for the different network loading problems. Moreover, we describe a decision support system with a graphical interface, based on these heuristics. The use of a graphical interface proves to give several advantages. Firstly, it makes the analysis and comparison of solutions easier for network planners. This in turn leads to a more intensive usage of the developed algorithms. Secondly, it helps in the analysis and development of local search heuristics themselves, because the effect of changes in algorithms can be visualized more effectively. Thirdly, in real-life applications one is not just interested in the optimal solution on the basis of a one-dimensional objective function. A graphical interface can give network planners a way to compare solutions on a set of criteria, yield insight into the robustness of solutions (sensitivity analysis), and even incorporate dynamics by analyzing the evolution of networks through time, for instance as a function of growing demand. In Chapter 5 we perform a computational study that yields answers to a number of network design questions.

In **Chapter 6** we look at the most basic version of the network loading problems as discussed in Chapters 5. These problems contain two types of restrictions. The flow balance equalities guarantee that traffic flows are routed from origin to destination. The edge capacity restrictions imply that sufficient capacity is installed on the links of the network to accommodate the associated flow on the link (edge). We study the model defined by a single edge constraint and prove that (non-trivial) facet-defining inequalities for the polytope asociated with a single edge capacity constraint are guaranteed to be facet-defining inequalities for the network loading polytope itself. We derive general characteristics of facet-defining inequalities for the edge capacity polytope, as well as some new classes of valid inequalities. We illustrate how valid inequalities for the standard 0–1 knapsack problem can be transformed into valid inequalities for the edge capacity polytope (and hence the network loading polytope) by mixed integer lifting. Mixed integer lifting as a theoretical procedure has been known for a long time (see for instance Wolsey [78]), however, most attention in the operations research literature on lifting procedures (both theoretical and in computational studies) has been on 0–1 lifting. Although mixed integer lifting is NP-hard for an arbitrary inequality, valid cover inequalities for knapsack polytopes can be lifted to valid inequalities for the edge capacity polytope in polynomial time. As a result of this analysis another characterization of c-strong inequalities as developed by Brockmüller, Günlück and Wolsey [20] is obtained. In a computational study we investigate the effect of the developed theory on the solvability of network loading problems.

Part I

Network Design Problems On Trees

multiplicity, diversity and complexity of these problems, developing exact algorithmic methods for this complete set of network loading problems is not only very hard but impossible. Therefore, we report on local search heuristics for the single-period network loading problems. Moreover, we describe a decision support system with graphical user interface, based on these heuristics. The use of a graphical user interface has several advantages. Firstly, it makes the analysis and design process more transparent for the user, with planners. This in turn leads to a more Secondly, it helps in the analysis and development of because the effect of changes in algorithmic of real-life applications one is not just into a multi-dimensional objective function, to compare solutions on a set of (sensitivity analysis), and even

Part I

Network Design Problems On Trees

Chapter 2

A Dynamic Programming Algorithm for the Local Access Telecommunication Network Expansion Problem

2.1 Introduction

The local access telecommunication network (LATN) is a tree network that connects user nodes to the switching center located in the root of the tree. Each user node typically represents a collection of individual users connected by an underlying network. Communication between user nodes of this and other LATNs is accomplished through the switching center located in the root of the tree. Traditional voice services require the same capacity in both directions. Therefore, instead of using directional traffic demand between pairs of nodes, we can assume that each user node has a demand that must be routed to the central switching center. The demand of a user node is usually measured in the required number of circuits needed between the node and the switching center, where each circuit requires one twisted copper cable in conventional copper networks.

Routing demand may be accomplished in two ways, viz. either by using dedicated cables from the user node to the switching center (via its unique path in the tree), or by routing the demand to a compression device called a concentrator, which is (to be) installed in a node of the network. A concentrator compresses all incoming low frequency signals (demand) into one outgoing high frequency (or optical) signal, which is then routed to the switching center. It is assumed that this outgoing signal either requires negligible capacity in the network, or is routed to the switching center via a dedicated line *not* belonging to the network. The costs of constructing such dedicated lines are included in the installation costs of the concentrators involved. In practice, a large variety of electronic devices are available to compress signals. For the problem at hand, these different technologies can simply be treated as concentrators with different capacities and operational costs.

Due to the introduction of new services, the increased intensity in the number of customers and the continuously growing utilization of telecommunication services, the existing capacity in the LATN may no longer suffice to accommodate this increasing demand. In that case, the objective is to expand cable capacity and/or install concentrators, and possibly reroute traffic demands from user nodes, in such a way that all demand is satisfied, and the costs of the network expansion plan are minimized. Hence, the key issue in Local Access Telecommunication Network Expansion Problem (which we will refer to as LATNEP henceforth) is to find an efficient trade-off between cable expansion and concentrator installation costs.

In Balakrishnan, Magnanti and Wong [9] the LATNEP with a specific cost structure is introduced. They mention that the problem is NP-hard and use Lagrangian relaxation, valid inequalities and preprocessing techniques to determine (near-)optimal solutions. For the special case where existing edge capacities are equal to zero (i.e. the *design* problem) and which only involves one uncapacitated concentrator type with a piecewise-linear and concave cost structure, they show that the problem is solvable in polynomial time (see also Barany, Edmonds and Wolsey [14]). Cho and Shaw [23] study LATNEP with a fixed charge cost structure, and solve the problem with a dynamic programming algorithm that is embedded in a column generation approach. For the design problem (i.e. existing capacities on edges are zero), their algorithm solves the problem in $\mathcal{O}(n^2B)$ time complexity and storage space, with n referring to the number of nodes in the tree, and B to an upper bound on concentrator capacity. For the more general expansion problem a similar approach is proposed, but the resulting algorithm is incorrect, since it may fail to find an optimal solution for certain problem instances.

In this chapter, based on Flippo, Koster, Kolen and van de Leensel [33], we show that LATNEP is weakly NP-hard. We present a dynamic programming algorithm for LATNEP that runs in $\mathcal{O}(nB^2)$ time and requires $\mathcal{O}(nB)$ storage space. Our algorithm can also handle more general cost structures for cable expansion and concentrator installation than previously considered in literature. These structures include non-convex and non-concave costs, which may be node and edge dependent. This allows us to incorporate many aspects occurring in practical planning problems, such as the availability of different electronic devices to perform multiplexing operations, the possibility to install multiple concentrators in a node, or even the demolition of concentrators, among others. The only assumption we impose is *decomposability*, i.e. total cable expansion (concentrator installation) costs are the sum of the individual expansion (installation) costs per edge (node).

Computational experiments indicate that the proposed algorithm is very efficient; networks up to 30 nodes (as mentioned in Cho and Shaw [23]) can be solved within fractions of seconds, whereas significantly larger instances up to 1000 nodes can be solved within (fractions of) seconds to minutes, depending on network structure and concentrator capacity B. Balakrishnan, Magnanti and Wong [9] test their method on three realistic problem instances from industry. The largest problem instance, a 41 node problem, could not be solved to optimality using their method (they report a 7% optimality gap after 15 minutes of running time). For confidentiality reasons we were not able to obtain the exact cost functions used by Balakrishnan, Magnanti and Wong [9], but since the running time

of our algorithm is basically *in*dependent of the cost structure (cf. Section 2.4), we could test our method on this problem instance using a variety of different cost functions. In all cases our algorithm solves the problem instance to optimality in less than a minute.

The remainder of this chapter is organized as follows. In Section 2.2 we give a detailed problem description and a mathematical formulation of LATNEP. In Section 2.3 we embed the problem into two parameterized families of subproblems, and we derive relations between the members of these families on which the dynamic programming algorithm is based. The algorithm itself is stated in Section 2.4, together with a proof of its correctness. Computational results are reported in Section 2.5.

2.2 Problem Description

Let $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ be the tree on which LATNEP is defined, with $\mathcal{V} = \{0, \ldots, n\}$ and $\mathcal{E} = \{1, \ldots, n\}$. We assume that both nodes and edges are numbered in a depth-first-search order. The predecessor of v is denoted by p_v. Hence, edge $e \in \mathcal{E}$ equals $\{p_v, v\}$, where $v \in \mathcal{V}$ and e have the same (numerical) label. For $v, w \in \mathcal{V}$, let $V(v, w)$ and $E(v, w)$ denote the nodes (endpoints included) and edges on the path from v to w in $\mathcal{T}$, respectively. For every node $v \in \mathcal{V}$ a traffic demand (load) r_v is given, and for every edge $e \in \mathcal{E}$ the existing capacity b_e is known. We assume that both the traffic demands and the existing capacities are integral. Next, for each $v \in \mathcal{V}$, a real-valued cost function K_v is given, where $K_v(k_v)$ specifies the concentrator costs that are involved when k_v is the amount of demand to be processed by a concentrator in node v (also referred to as the load on node v). Likewise, for every edge $e \in \mathcal{E}$ a real-valued cost function L_e is given, where $L_e(\ell_e)$ specifies the cable costs that are involved when a demand (load) of ℓ_e is to be transferred over edge e.

Due to the generality of these cost structures, a variety of problem characteristics can be taken into account as special cases. The situation where a concentrator with capacity $\hat{b}_v$ is already operational in node v for instance, can be accounted for by setting $K_v(k_v)$ equal to the costs of installing and operating a new or supplementary concentrator if the (planned) load k_v exceeds the current capacity $\hat{b}_v$, and zero otherwise. Possible demolition costs of such an existing concentrator in v could thereby also be included in $K_v(k_v)$. Similarly, the situation where it is allowed to install multiple concentrators in a node can be handled by the model. Finally, the fixed charge cost structures that are considered in Balakrishnan, Magnanti and Wong [9] and Cho and Shaw [23] can be accounted for. In both studies concentrators are available in different types. Every concentrator of type t has a given capacity $\hat{b}^t$, and (node dependent) fixed and variable installation costs $\hat{F}_v^t$ and $\hat{c}_v^t$ respectively. (On close inspection, the variable costs in Cho and Shaw [23] are assumed *in*dependent of the concentrator type t). This situation is recovered from our cost structure by setting $K_v(k_v)$ equal to $\min_t\{\hat{F}_v^t + k_v \cdot \hat{c}_v^t \mid \hat{b}^t \geq k_v\}$ for $k_v > 0$, and zero otherwise. Note that this cost structure therefore incorporates the variety of different concentrating devices which may be available in practical situations. The edge costs that are considered in these papers have a similar structure. For every edge $e \in \mathcal{E}$ an existing capacity b_e, and (edge dependent) fixed and variable expansion costs F_e and c_e are

defined; cable expansion costs are then obtained by setting $L_e(\ell_e)$ equal to $F_e + (\ell_e - b_e) \cdot c_e$ for $\ell_e > b_e$, and zero otherwise. For an example of the LATNEP problem, we refer to Figure:1.5 and its discussion.

The main issue in LATNEP is to decide for each node $v \in \mathcal{V}$ whether to route its demand to the switching center (which we refer to as the concentrator in the root node) via its unique path in the tree, or to route it to a node $w \in \mathcal{V} \setminus \{0\}$ in which a concentrator must then be present or installed to transmit all incoming load to the switching center of $\mathcal{T}$ via a dedicated line. If the load of node v is routed to a concentrator in node w, we say that v "homes on" w (cf. Balakrishnan, Magnanti and Wong [9]). Note that we do allow *backfeed*, i.e. a node can also home on nodes other than those on the unique path to the root of the tree.

Apart from routing restrictions due to technical constraints, network planners often impose extra restrictions on the layout of telecommunication networks. Some of these restrictions are considered to be economically sensible, while other restrictions are considered practical for operational convenience regarding maintenance and repair. For LATNEP, the following restrictions should be incorporated (these restrictions are the same as proposed by Balakrishnan, Magnanti and Wong [9], Cho and Shaw [23], Shulman and Vachani [71] and Jack, Kai and Shulman [50]).

1. **Single level concentration:** demand is concentrated at most once before reaching the switching center in the root of the tree;

2. **Nonbifurcated routing:** for each user node its entire demand is processed by a single concentrator (possibly at the root);

3. **Contiguity condition:** if a node v homes on a concentrator in node w, then all nodes on the path from v to w home on w.

Condition 1 reflects guidelines of network planners who, given the current relative costs of cable expansion and concentrator installation, consider multiple levels of concentration to be uneconomical. The compressed (high frequency) signal is assumed to be routed from the concentrator to the switching center using a dedicated line *not* belonging to the network. Conditions 2 and 3 are enforced to ensure operational convenience of maintenance and repair (for a detailed description of these conditions, we refer to Balakrishnan, Magnanti and Wong [9]). Note that due to these routing restrictions, once it is known for each node v on which node w it homes, the complete configuration of the network is known. Therefore, we define

$$
\begin{array}{lcll}
x_{uw} & = & \begin{cases} 1 & \text{if node } u \text{ homes on node } w \\ 0 & \text{otherwise} \end{cases} & (u, w \in \mathcal{V}) \\
k_v & = & \text{the load to be processed by a concentrator in node } v & (v \in \mathcal{V}) \\
\ell_e & = & \text{the load to be transferred over edge } e & (e \in \mathcal{E})
\end{array}
$$

Then LATNEP reads:

$$\begin{aligned}
\min \quad & \textstyle\sum_{w\in\mathcal{V}} K_w(k_w) + \sum_{e\in\mathcal{E}} L_e(\ell_e) && && (2.1)\\
\text{s.t.} \quad & x_{00} = 1 && && (2.2)\\
& \textstyle\sum_{w\in\mathcal{V}} x_{uw} = 1 && \forall u \in \mathcal{V} && (2.3)\\
& x_{u'w} \geq x_{uw} && \forall u, u', w \in \mathcal{V} : u' \in V(u,w) && (2.4)\\
& k_w = \textstyle\sum_{u\in\mathcal{V}} r_u \cdot x_{uw} && \forall w \in \mathcal{V} && (2.5)\\
& \ell_e = \textstyle\sum_{u,w\in\mathcal{V}: e\in E(u,w)} r_u \cdot x_{uw} && \forall e \in \mathcal{E} && (2.6)\\
& k_w \leq B && \forall w \in \mathcal{V} && (2.7)\\
& x_{uw} \in \{0,1\},\ k_w \geq 0,\ \ell_e \geq 0 && \forall u, w \in \mathcal{V},\ \forall e \in \mathcal{E} && (2.8)
\end{aligned}$$

The objective function in (2.1) defines the total costs that follow from the network expansion program (x, k, ℓ). As can be seen from its formulation, it propagates the decomposability assumption on the costs. As for concentrator costs in node w, it follows from the load k_w whether or not a concentrator should be installed in node w, and the costs $K_w(k_w)$ can be defined accordingly to describe this situation correctly. Furthermore, existing edge capacities can be incorporated as indicated before. Constraint (2.2) states that a concentrator (the switching center) is installed in the root node. Constraint (2.3) implies that every node homes on exactly one node (and thereby ensures the nonbifurcated routing condition), whereas (2.4) enforces the contiguity condition. Note that (2.4) contains a lot of redundancy. However, since the model is only used to communicate the problem and prove the validity of our dynamic programming approach, efficiency in the number of constraints is not an issue here. Constraints (2.5) and (2.6) merely define the resulting loads on the nodes and edges, respectively. Without loss of generality we assume in (2.7) that for any given $w \in \mathcal{V}$, a uniform bound B exists that restricts the sum of the loads of all nodes homing on w to B. In practical situations this upper bound is given by the maximum of the capacities of the available concentrator types, or in the case of an uncapacitated concentrator by the sum of all the loads in the tree $\mathcal{T}$. The integrality and non-negativity constraints in (2.8) complete the formulation. Although different formulations are presented for LATNEP in the literature (Balakrishnan, Magnanti and Wong [9] and Cho and Shaw [23] include variables to represent the cable expansion decision as well as the installation of concentrators) they describe the same problem.

Following standard practice, we define the minimum over an empty set to be ∞, and the summation over an empty set to be zero. The indicator function for a logical expression A is denoted by $1_{[A]}$, which equals 1 or 0, depending on whether A evaluates to true or false. Finally, we denote the set of feasible solutions by $\mathcal{F}$, hence

$$\mathcal{F} = \{(x, k, \ell) \mid (x, k, \ell) \text{ satisfies (2.2)–(2.8)}\}.$$

2.3 Parameterizations for LATNEP

In this section we introduce two parameterized families of subproblems. These subproblems are defined on certain subtrees of $\mathcal{T}$, which are the subject of Subsection 2.3.1. The mathematical formulations for the parameterized problems is stated in Subsection 2.3.2, as well as a motivation for their definition and an intuitive explanation of how the dynamic programming algorithm works. In order to rigorously prove the correctness of the algorithm, we need to state some relevant relationships between the subproblems involved. These relations are twofold. On the one hand, we prove that solutions to a given subproblem are also solutions to the subproblems of which it is composed. This *downward compatibility of solutions* will be the subject of Section 2.3.3. On the other hand, we will show that solutions of certain subproblems can be combined to form a solution of the encapsulating subproblem. This *upward compatibility of solutions* will be discussed in Section 2.3.4. Based on these results, the final relations between the various subproblems that are used by our dynamic programming algorithm, are stated in Section 2.3.5.

2.3.1 Defining Subtrees $T[v,i]$ of Tree $\mathcal{T}$

The subtrees we employ were introduced by Johnson and Niemi [51] to efficiently solve tree knapsack problems and tree partitioning problems. Let d_v be the number of children (successors) of node v in $\mathcal{T}$, and $D_v = \{s_v^1, \ldots, s_v^{d_v}\}$ the set of its children, with s_v^i the i^{th} child of v. For $v \in \mathcal{V}$ and $0 \leq i \leq d_v$ we define the subtree $T[v,i]$, which is induced by node v, its first i children $\{s_v^1, \ldots, s_v^i\}$ and all successors of these children (see Johnson and Niemi [51]). For example, in the tree of Figure 1.5, $T[4,1]$ is given by the subtree defined on the nodes $4,5,6,7$, whereas $T[4,2]$ is given by the subtree defined on the nodes $4,5,6,7,8$. Note that $T[0,d_0]$ is the complete tree $\mathcal{T}$, and $T[v,0]$ is the subtree of $\mathcal{T}$ consisting only of the node v. Also observe, that for $v \in \mathcal{V}$ and $1 \leq i \leq d_v$, the tree $T[v,i]$ consists of the subtrees $T[v,i-1]$ and $T[s_v^i, d_{s_v^i}]$, together with the edge $\{v, s_v^i\}$. Finally, let $V[v,i]$ be the node set of $T[v,i]$.

2.3.2 Defining Subproblems on Subtrees

Given an arbitrary subtree $T = (V,E)$ of $\mathcal{T}$ and an arbitrary solution $(x,k,\ell) \in \mathcal{F}$, we define the costs of subtree T for the solution (x,k,ℓ) by:

$$C(x,k,\ell \mid T) \;=\; \textstyle\sum_{w \in V} K_w(k_w) + \sum_{e \in E} L_e(\ell_e) \tag{2.9}$$

Note that for $w \in V$, the variable k_w also contains the load from nodes that do *not* belong to T, but that do home on w. Similarly, ℓ_e may contain load from outside (inside) T that is transferred over e to a concentrator inside (outside) T. In the first family of subproblems we restrict ourselves to the case in which the root of the subtree homes on a node *in*side

the subtree: for (v,i) with $v \in \mathcal{V}$, $0 \leq i \leq d_v$ and $0 \leq s \leq \min(B - r_v, \sum_{u \notin V[v,i]} r_u)$ let

$$
\begin{aligned}
g(v,i,s) \quad = \quad & \min \; C(x,k,\ell \mid T[v,i]) && (2.10)\\
& \text{s.t.} \; x_{vw} = 0 \qquad \forall w \notin V[v,i] && (2.11)\\
& \textstyle\sum_{u \notin V[v,i]} \sum_{w \in V[v,i]} r_u \cdot x_{uw} = s && (2.12)\\
& (x,k,\ell) \in \mathcal{F} && (2.13)
\end{aligned}
$$

So, $g(v,i,s)$ represents the minimal costs of subtree $T[v,i]$ among all solutions $(x,k,\ell) \in \mathcal{F}$ for which v homes within $T[v,i]$, and the total demand from nodes *not* in $T[v,i]$ homing within $T[v,i]$ equals s. Note that by contiguity, this load s must home on the same node as node v does (say node $\tilde{w}$), which explains the upper bound on s. Moreover, in order to determine the concentrator costs in node $\tilde{w}$, the complete load which has to be processed by this concentrator has to be known. Since part of this complete load may be due to demand from nodes outside $T[v,i]$, the parameter s is incorporated in the parameterization. Finally, note that $g(0,d_0,0)$ is equivalent to LATNEP, hence, this is the problem we ultimately want to solve.

In the second family of subproblems we restrict ourselves to the case in which the root of the subtree homes on a node *out*side the subtree: for (v,i) with $v \in \mathcal{V}$, $0 \leq i \leq d_v$ and $r_v \leq r \leq \min(B, \sum_{u \in V[v,i]} r_u)$ we define

$$
\begin{aligned}
h(v,i,r) \quad = \quad & \min \; C(x,k,\ell \mid T[v,i]) && (2.14)\\
& \text{s.t.} \; x_{vw} = 0 \qquad \forall w \in V[v,i] && (2.15)\\
& \textstyle\sum_{u \in V[v,i], w \notin V[v,i]} r_u \cdot x_{uw} = r && (2.16)\\
& (x,k,\ell) \in \mathcal{F} && (2.17)
\end{aligned}
$$

So, $h(v,i,r)$ represents the minimal costs of subtree $T[v,i]$ among all solutions $(x,k,\ell) \in \mathcal{F}$ for which node v does *not* home within $T[v,i]$ and the total demand of nodes in $T[v,i]$ homing outside $T[v,i]$ equals r. Again, note that by contiguity this load r must home on the same node as node v.

Our dynamic programming algorithm operates in a bottom-to-top kind of fashion as follows. Suppose that in the example of Figure 1.5 subtree $T[4,2]$ is under consideration for the calculation of $g(4,2,s)$. Since $T[4,2]$ is composed of the two subtrees $T[4,1]$ and $T[8,0]$ (along with edge 8), the general idea is to obtain an optimal solution to the former by combining optimal solutions of the latter two. Since in $g(4,2,s)$ node 4 must home within $T[4,2]$, two situations may arise. On the one hand, it may be optimal to combine an optimal solution of $T[8,0]$ with node 8 homing on some node in $T[8,0]$ (necessarily node 8 in this example), with an optimal solution of $T[4,1]$ with node 4 homing on some node in $T[4,1]$. In this case the former of the two optimal solutions is an optimal solution for $g(8,0,0)$ and the latter is an optimal solution for $g(4,1,s)$. On the other hand it may be optimal to combine optimal solutions of $T[8,0]$ and $T[4,1]$ with nodes 4 and 8 both homing on the same node w. In case $w \in V[4,1]$, the load from $T[8,0]$ is transferred to

$T[4,1]$ via edge 8. Since the resulting costs depend on the load that is transferred over edge 8, it is necessary to know how much load is actually involved. Let α denote the load over edge 8, then the solution for $g(4,2,s)$ can be obtained by combining solutions $h(8,0,\alpha)$ and $g(4,1,s+\alpha)$. Similarly, if $w \in V[8,0]$ then the load is transferred from $T[4,1]$ to $T[8,0]$ via edge 8, and the solution for $g(4,2,s)$ can be obtained by combining solutions $g(8,0,s+\alpha)$ and $h(4,1,\alpha)$. For the calculation of $h(4,2,r)$ several cases can be distinguished in a similar manner. These ideas are now formalized in the sequel for the general case in which we combine solutions of the subtrees $T[v,i-1]$ and $T[s_v^i, d_{s_v^i}]$ to obtain a solution for the subtree $T[v,i]$.

It is important to note that $g(v,i,s)$ and $h(v,i,r)$ may not have feasible solutions, since constraints (2.12) and (2.16) may be impossible to satisfy. For example, if the demand r_v is even for all user nodes v, then $g(v,i,s)$ and $h(v,i,r)$ are infeasible if the parameters s and r are odd. We will return to this issue on several occasions in the sequel; firstly in Subsection 2.3.5, when the recursive relations on which the dynamic programming algorithm is based, are discussed, and secondly in Section 2.4, when the correctness of the algorithm is considered. The following lemmas, which are implied by the contiguity condition, will prove to be helpful in the subsequent analysis. We confine to stating the proof of the first lemma, as the remaining proofs are similar.

Lemma 2.3.1 *Consider (v,i) with $v \in \mathcal{V}$ and $1 \leq i \leq d_v$. Let $(x,k,\ell) \in \mathcal{F}$ be a solution for which $x_{vw} = 0$ for all $w \notin V[s_v^i, d_{s_v^i}]$. Then $x_{uw} = 0$ for all $u,w \in \mathcal{V}$ with $v \in V(u,w)$ and $w \notin V[s_v^i, d_{s_v^i}]$.*

Proof. Let $u,w \in \mathcal{V}$ be such that $v \in V(u,w)$ and $w \notin V[s_v^i, d_{s_v^i}]$. Suppose that $x_{uw} = 1$. Since $v \in V(u,w)$ it follows by (2.4) and (2.8) that $x_{vw} = 1$, a clear contradiction. ■

Lemma 2.3.2 *Consider (v,i) with $v \in \mathcal{V}$ and $1 \leq i \leq d_v$. Let $(x,k,\ell) \in \mathcal{F}$ be a solution for which $x_{vw} = 0$ for all $w \notin V[v,i-1]$. Then $x_{uw} = 0$ for all $u,w \in \mathcal{V}$ with $v \in V(u,w)$ and $w \notin V[v,i-1]$.*

Lemma 2.3.3 *Consider (v,i) with $v \in \mathcal{V}$ and $0 \leq i \leq d_v$. Let $(x,k,\ell) \in \mathcal{F}$ be a solution for which $x_{vw} = 0$ for all $w \notin V[v,i]$. Then $x_{uw} = 0$ for all $u,w \in \mathcal{V}$ with $v \in V(u,w)$ and $w \notin V[v,i]$.*

Lemma 2.3.4 *Consider (v,i) with $v \in \mathcal{V}$ and $0 \leq i \leq d_v$. Let $(x,k,\ell) \in \mathcal{F}$ be a solution for which $x_{vw} = 0$ for all $w \in V[v,i]$. Then $x_{uw} = 0$ for all $u,w \in \mathcal{V}$ with $v \in V(u,w)$ and $w \in V[v,i]$.*

2.3.3 Downward Compatibility of Solutions

Consider a pair (v,i) with $v \in \mathcal{V}$ and $1 \leq i \leq d_v$. Below we will show that a feasible solution for $g(v,i,s)$ is feasible for either both $g(s_v^i, d_{s_v^i}, s+\alpha)$ and $h(v,i-1,\alpha)$ for some

α with $r_v \le \alpha \le B - s - r_{s^i_v}$, feasible for both $h(s^i_v, d_{s^i_v}, \alpha)$ and $g(v, i-1, s+\alpha)$ for some α with $r_{s^i_v} \le \alpha \le B - s - r_v$, or feasible for both $g(s^i_v, d_{s^i_v}, 0)$ and $g(v, i-1, s)$. To not withdraw the reader's attention from the main results, the proof of Lemma 2.3.5 is listed at the end of the subsection. The proofs of the other lemmas are analogous.

Lemma 2.3.5 *Consider (v,i) with $v \in \mathcal{V}$ and $1 \le i \le d_v$. Let $(x,k,\ell) \in \mathcal{F}$ be a feasible solution for $g(v,i,s)$ with $x_{vw} = 0$ for all $w \notin V[s^i_v, d_{s^i_v}]$. If α satisfies $\ell_{s^i_v} = s + \alpha$, then*

(i). (x,k,ℓ) is feasible for both $g(s^i_v, d_{s^i_v}, s+\alpha)$ and $h(v, i-1, \alpha)$;

(ii). $C(x,k,\ell \mid T[v,i]) = C(x,k,\ell \mid T[s^i_v, d_{s^i_v}]) + C(x,k,\ell \mid T[v,i-1]) + L_{s^i_v}(s+\alpha)$;

(iii). $r_v \le \alpha \le B - s - r_{s^i_v}$.

Lemma 2.3.6 *Consider (v,i) with $v \in \mathcal{V}$ and $1 \le i \le d_v$. Let $(x,k,\ell) \in \mathcal{F}$ be a feasible solution for $g(v,i,s)$ with $x_{s^i_v w} = 0$ for all $w \notin V[v, i-1]$. If α satisfies $\ell_{s^i_v} = \alpha$, then*

(i). (x,k,ℓ) is feasible for both $h(s^i_v, d_{s^i_v}, \alpha)$ and $g(v, i-1, s+\alpha)$;

(ii). $C(x,k,\ell \mid T[v,i]) = C(x,k,\ell \mid T[s^i_v, d_{s^i_v}]) + C(x,k,\ell \mid T[v,i-1]) + L_{s^i_v}(\alpha)$;

(iii). $r_{s^i_v} \le \alpha \le B - s - r_v$.

Lemma 2.3.7 *Consider (v,i) with $v \in \mathcal{V}$ and $1 \le i \le d_v$. Let $(x,k,\ell) \in \mathcal{F}$ be a feasible solution for $g(v,i,s)$ with $x_{vw} = 0$ for all $w \notin V[v,i-1]$ and $x_{s^i_v w} = 0$ for all $w \notin V[s^i_v, d_{s^i_v}]$. Then*

(i). (x,k,ℓ) is feasible for both $g(s^i_v, d_{s^i_v}, 0)$ and $g(v, i-1, s)$;

(ii). $C(x,k,\ell \mid T[v,i]) = C(x,k,\ell \mid T[s^i_v, d_{s^i_v}]) + C(x,k,\ell \mid T[v,i-1]) + L_{s^i_v}(0)$.

The following lemmas indicate that a feasible solution for $h(v,i,r)$ is a feasible solution to either both $h(s^i_v, d_{s^i_v}, \alpha)$ and $h(v, i-1, r-\alpha)$ for some α with $r_{s^i_v} \le \alpha \le r - r_v$, or feasible for both $g(s^i_v, d_{s^i_v}, 0)$ and $h(v, i-1, r)$.

Lemma 2.3.8 *Consider (v,i) with $v \in \mathcal{V}$ and $1 \le i \le d_v$. Let $(x,k,\ell) \in \mathcal{F}$ be a feasible solution for $h(v,i,r)$ with $x_{s^i_v w} = 0$ for all $w \in V[s^i_v, d_{s^i_v}]$. If α satisfies $\ell_{s^i_v} = \alpha$, then*

(i). (x,k,ℓ) is feasible for both $h(s^i_v, d_{s^i_v}, \alpha)$ and $h(v, i-1, r-\alpha)$;

(ii). $C(x,k,\ell \mid T[v,i]) = C(x,k,\ell \mid T[s^i_v, d_{s^i_v}]) + C(x,k,\ell \mid T[v,i-1]) + L_{s^i_v}(\alpha)$;

(iii). $r_{s^i_v} \le \alpha \le r - r_v$.

Lemma 2.3.9 *Consider (v,i) with $v \in \mathcal{V}$ and $1 \le i \le d_v$. Let $(x,k,\ell) \in \mathcal{F}$ be a feasible solution for $h(v,i,r)$ with $x_{s_v^i w} = 0$ for all $w \notin V[s_v^i, d_{s_v^i}]$. Then*

(i). (x,k,ℓ) is feasible for both $g(s_v^i, d_{s_v^i}, 0)$ and $h(v,i-1,r)$;

(ii). $C(x,k,\ell \mid T[v,i]) = C(x,k,\ell \mid T[s_v^i, d_{s_v^i}]) + C(x,k,\ell \mid T[v,i-1]) + L_{s_v^i}(0)$.

Proof. (of Lemma 2.3.5).

(i). For $u \in V[s_v^i, d_{s_v^i}]$ and $w \notin V[s_v^i, d_{s_v^i}]$ it follows directly from Lemma 2.3.1 that $x_{uw} = 0$. Moreover, since $s + \alpha = \ell_{s_v^i}$ it follows that

$$\begin{aligned} s+\alpha &= \textstyle\sum_{u,w \in \mathcal{V}: s_v^i \in E(u,w)} r_u \cdot x_{uw} \\ &= \textstyle\sum_{u \notin V[s_v^i, d_{s_v^i}], w \in V[s_v^i, d_{s_v^i}]} r_u \cdot x_{uw} + \sum_{u \in V[s_v^i, d_{s_v^i}], w \notin V[s_v^i, d_{s_v^i}]} r_u \cdot x_{uw} \\ &= \textstyle\sum_{u \notin V[s_v^i, d_{s_v^i}], w \in V[s_v^i, d_{s_v^i}]} r_u \cdot x_{uw} \end{aligned}$$

Consequently, (x,k,ℓ) is feasible for $g(s_v^i, d_{s_v^i}, s+\alpha)$. To show the feasibility of $h(v,i-1,\alpha)$, note that $x_{vw} = 0$ follows immediately from the condition in the lemma. Next,

$$\begin{aligned} \sum_{\substack{u \in V[v,i-1] \\ w \notin V[v,i-1]}} r_u \cdot x_{uw} &= \sum_{\substack{u \in V[v,i-1] \\ w \notin V[v,i]}} r_u \cdot x_{uw} + \sum_{\substack{u \in V[v,i-1] \\ w \in V[s_v^i, d_{s_v^i}]}} r_u \cdot x_{uw} \\ &= 0 + \sum_{\substack{u \notin V[s_v^i, d_{s_v^i}] \\ w \in V[s_v^i, d_{s_v^i}]}} r_u \cdot x_{uw} - \sum_{\substack{u \notin V[v,i] \\ w \in V[s_v^i, d_{s_v^i}]}} r_u \cdot x_{uw} \\ &= (\ell_{s_v^i} - \sum_{\substack{u \in V[s_v^i, d_{s_v^i}] \\ w \notin V[s_v^i, d_{s_v^i}]}} r_u \cdot x_{uw}) \\ &\quad -(\sum_{\substack{u \notin V[v,i] \\ w \in V[v,i]}} r_u \cdot x_{uw} - \sum_{\substack{u \notin V[v,i] \\ w \in V[v,i-1]}} r_u \cdot x_{uw}) \\ &= (s+\alpha-0) - (s-0) \\ &= \alpha \end{aligned}$$

where the second and fourth equality hold by Lemma 2.3.1.

(ii). Follows directly from $\ell_{s_v^i} = s + \alpha$ and the decomposability of the costs.

(iii). Let $\tilde{w} \in V[s_v^i, d_{s_v^i}]$ be the node for which $x_{v\tilde{w}} = 1$. Then

$$r_v = r_v \cdot x_{v\tilde{w}} \le \textstyle\sum_{u \in V[v,i-1], w \notin V[v,i-1]} r_u \cdot x_{uw} = \alpha$$

By contiguity, $x_{s_v^i \tilde{w}} = 1$, hence (2.5) and (2.7) yield

$$B \ge \textstyle\sum_{u \in \mathcal{V}} r_u \cdot x_{u\tilde{w}} = \sum_{u \notin V[s_v^i, d_{s_v^i}]} r_u \cdot x_{u\tilde{w}} + \sum_{u \in V[s_v^i, d_{s_v^i}]} r_u \cdot x_{u\tilde{w}} \ge s + \alpha + r_{s_v^i}$$

This completes the proof. ∎

2.3.4 Upward Compatibility of Solutions

Consider a pair (v,i) with $v \in \mathcal{V}$ and $1 \le i \le d_v$. Below we will show that some feasible solutions for some problems on $T[v,i]$, $T[v,i-1]$ and $T[s^i_v, d_{s^i_v}]$ can be combined to obtain solutions which are simultaneously feasible on all three subtrees. Once again the proof of the first lemma is listed at the end of the subsection; the other lemmas can be proven similarly.

Lemma 2.3.10 *Consider (v,i) with $v \in \mathcal{V}$ and $1 \le i \le d_v$. Let*

- *(x^1, k^1, ℓ^1) be a feasible solution for $g(s^i_v, d_{s^i_v}, s+\alpha)$,*
- *(x^2, k^2, ℓ^2) be a feasible solution for $h(v, i-1, \alpha)$, and*
- *(x^3, k^3, ℓ^3) be a feasible solution for $g(v,i,s)$.*

Let $w^* \in V[s^i_v, d_{s^i_v}]$ be such that $x^1_{s^i_v w^*} = 1$. Since the value of $h(v,i-1,\alpha)$ does not depend on the homing node of v (which is not in $T[v,i-1]$), assume w.l.o.g. that $x^2_{vw^*} = 1$. Define the composite solution (x^4, k^4, ℓ^4) by

$$x^4_{uw} = \begin{cases} x^1_{uw} & \text{if } u \in V[s^i_v, d_{s^i_v}] \\ x^2_{uw} & \text{if } u \in V[v, i-1] \\ x^3_{uw} & \text{if } u \notin V[v,i], w \notin V[v,i] \\ 0 & \text{if } u \notin V[v,i], w \in V[v,i] \setminus \{w^*\} \\ \sum_{u' \in V[v,i]} x^3_{uu'} & \text{if } u \notin V[v,i], w = w^* \end{cases}$$

and (k^4, ℓ^4) according to (2.5)–(2.6) for $x = x^4$. Then (x^4, k^4, ℓ^4) is a feasible solution for $g(s^i_v, d_{s^i_v}, s+\alpha)$, $h(v, i-1, \alpha)$ and $g(v,i,s)$, with

$$\begin{aligned} C(x^4,k^4,\ell^4 \mid T[s^i_v, d_{s^i_v}]) &= C(x^1,k^1,\ell^1 \mid T[s^i_v, d_{s^i_v}]) \\ C(x^4,k^4,\ell^4 \mid T[v,i-1]) &= C(x^2,k^2,\ell^2 \mid T[v,i-1]) \\ C(x^4,k^4,\ell^4 \mid T[v,i]) &= C(x^1,k^1,\ell^1 \mid T[s^i_v, d_{s^i_v}]) + C(x^2,k^2,\ell^2 \mid T[v,i-1]) \\ &\quad + L_{s^i_v}(s+\alpha) \end{aligned}$$

Lemma 2.3.11 *Consider (v,i) with $v \in \mathcal{V}$ and $1 \le i \le d_v$. Let*

- *(x^1, k^1, ℓ^1) be a feasible solution for $h(s^i_v, d_{s^i_v}, \alpha)$,*
- *(x^2, k^2, ℓ^2) be a feasible solution for $g(v, i-1, s+\alpha)$, and*
- *(x^3, k^3, ℓ^3) be a feasible solution for $g(v,i,s)$.*

Let $w^* \in V[v,i-1]$ be such that $x^2_{vw^*} = 1$. W.l.o.g. assume that $x^1_{s^i_v w^*} = 1$. Define the composite solution (x^4,k^4,ℓ^4) as in Lemma 2.3.10 and (k^4,ℓ^4) according to (2.5)-(2.6) for $x = x^4$. Then (x^4,k^4,ℓ^4) is a feasible solution for $h(s^i_v, d_{s^i_v}, \alpha)$ and $g(v,i-1,s+\alpha)$ and $g(v,i,s)$, with

$$\begin{aligned} C(x^4,k^4,\ell^4 \mid T[s^i_v, d_{s^i_v}]) &= C(x^1,k^1,\ell^1 \mid T[s^i_v, d_{s^i_v}]) \\ C(x^4,k^4,\ell^4 \mid T[v,i-1]) &= C(x^2,k^2,\ell^2 \mid T[v,i-1]) \\ C(x^4,k^4,\ell^4 \mid T[v,i]) &= C(x^1,k^1,\ell^1 \mid T[s^i_v, d_{s^i_v}]) + C(x^2,k^2,\ell^2 \mid T[v,i-1]) \\ &\quad + L_{s^i_v}(\alpha) \end{aligned}$$

Lemma 2.3.12 *Consider (v,i) with $v \in \mathcal{V}$ and $1 \le i \le d_v$. Let*

- *(x^1,k^1,ℓ^1) be a feasible solution for $g(s^i_v, d_{s^i_v}, 0)$,*
- *(x^2,k^2,ℓ^2) be a feasible solution for $g(v,i-1,s)$, and*
- *(x^3,k^3,ℓ^3) be a feasible solution for $g(v,i,s)$.*

Let $w^* \in V[v,i-1]$ be such that $x^2_{vw^*} = 1$. Define the composite solution (x^4,k^4,ℓ^4) as in Lemma 2.3.10 and (k^4,ℓ^4) according to (2.5)-(2.6) for $x = x^4$. Then (x^4,k^4,ℓ^4) is a feasible solution for $g(s^i_v, d_{s^i_v}, 0)$ and $g(v,i-1,s)$ and $g(v,i,s)$, for which the same cost relations hold as in Lemma 2.3.11 with $\alpha = 0$.

Lemma 2.3.13 *Consider (v,i) with $v \in \mathcal{V}$ and $1 \le i \le d_v$. Let*

- *(x^1,k^1,ℓ^1) be a feasible solution for $h(s^i_v, d_{s^i_v}, \alpha)$,*
- *(x^2,k^2,ℓ^2) be a feasible solution for $h(v,i-1,r-\alpha)$, and*
- *(x^3,k^3,ℓ^3) be a feasible solution for $h(v,i,r)$.*

Let $w^* \notin V[v,i]$ be such that $x^3_{vw^*} = 1$. W.l.o.g. assume that $x^1_{s^i_v w^*} = x^2_{vw^*} = 1$. Define the composite solution (x^4,k^4,ℓ^4) by

$$x^4_{uw} = \begin{cases} x^1_{uw} & \text{if } u \in V[s^i_v, d_{s^i_v}] \\ x^2_{uw} & \text{if } u \in V[v,i-1] \\ x^3_{uw} & \text{if } u \notin V[v,i] \end{cases}$$

and (k^4,ℓ^4) according to (2.5)-(2.6) for $x = x^4$. Then (x^4,k^4,ℓ^4) is a feasible solution for $h(s^i_v, d_{s^i_v}, \alpha)$ and $h(v,i-1,r-\alpha)$ and $h(v,i,r)$, for which the same cost relations hold as in Lemma 2.3.11.

Lemma 2.3.14 *Consider (v,i) with $v \in \mathcal{V}$ and $1 \leq i \leq d_v$. Let*

- *(x^1, k^1, ℓ^1) be a feasible solution for $g(s_v^i, d_{s_v^i}, 0)$,*
- *(x^2, k^2, ℓ^2) be a feasible solution for $h(v, i-1, r)$, and*
- *(x^3, k^3, ℓ^3) be a feasible solution for $h(v, i, r)$.*

Let $w^* \notin V[v,i]$ be such that $x^3_{vw^*} = 1$. W.l.o.g. assume that $x^2_{vw^*} = 1$. Define the composite solution (x^4, k^4, ℓ^4) as in Lemma 2.3.13 and (k^4, ℓ^4) according to (2.5)–(2.6) for $x = x^4$. Then (x^4, k^4, ℓ^4) is a feasible solution for $g(s_v^i, d_{s_v^i}, 0)$ and $h(v, i-1, r)$ and $h(v, i, r)$, for which the same cost relations hold as in Lemma 2.3.11 with $\alpha = 0$.

Proof. (of Lemma 2.3.10)

To show feasibility, we check the individual constraints.

ad (2.2) For $v \neq 0$, $x^4_{00} = x^3_{00} = 1$; for $v = 0$, $x^4_{00} = x^2_{00} = 1$;

ad (2.3) For $u \in V[v,i]$ the result follows immediately from the feasibility of the individual solutions. For $u \notin V[v,i]$:

$$\begin{aligned} \textstyle\sum_{w \in \mathcal{V}} x^4_{uw} &= \textstyle\sum_{w \notin V[v,i]} x^4_{uw} + \sum_{w \in V[v,i] \setminus \{w^*\}} x^4_{uw} + x^4_{uw^*} \\ &= \textstyle\sum_{w \notin V[v,i]} x^3_{uw} + 0 + \sum_{w \in V[v,i]} x^3_{uw} = 1 \end{aligned}$$

ad (2.4) Consider a triple (u, u'', w) with $u'' \in V(u,w)$. If u and u'' are both in $V[s_v^i, d_{s_v^i}]$, both in $V[v, i-1]$ or neither in $V[v,i]$ then $x^4_{u''w} \geq x^4_{uw}$ follows directly from the contiguity of the individual solutions. On the other hand, if u and u'' are located in different subtrees, we can distinguish six cases.

Suppose $u \notin V[v,i]$ and $u'' \in V[s_v^i, d_{s_v^i}]$. From $u'' \in V(u,w)$ it follows that $w \in V[s_v^i, d_{s_v^i}]$. If $w \neq w^*$, $x^4_{uw} = 0$ and the result follows immediately. If $w = w^*$, then from $u'' \in V(u,w)$ and $u'' \in V[s_v^i, d_{s_v^i}]$ it also follows that $u'' \in V(s_v^i, w)$. Hence, $x^4_{u''w} = x^1_{u''w} \geq x^1_{s_v^i w} = 1$, and the result follows. The remaining five cases can be shown similarly.

ad (2.5)–(2.6) These constraints are satisfied by definition.

ad (2.7) The result immediately follows once we have shown that for all $w \in \mathcal{V}$: $k^4_w = k^j_w$ for $j = 1, 2, 3$ depending on whether $w \in V[s_v^i, d_{s_v^i}]$, $w \in V[v, i-1]$ or $w \notin V[v,i]$; the result then follows from the feasibility of k^j_w in (2.7) ($j = 1, 2, 3$). First consider the case $w = w^*$. Then

$$\begin{aligned} k^4_w &= \textstyle\sum_{u \in \mathcal{V}} r_u \cdot x^4_{uw} \\ &= \textstyle\sum_{u \in V[s_v^i, d_{s_v^i}]} r_u \cdot x^1_{uw} + \sum_{u \in V[v,i-1]} r_u \cdot x^2_{uw} + \sum_{u \notin V[v,i], u' \in V[v,i]} r_u \cdot x^3_{uu'} \\ &= \textstyle\sum_{u \in V[s_v^i, d_{s_v^i}]} r_u \cdot x^1_{uw} + \alpha + s = k^1_w \end{aligned}$$

Next, consider the case $w \neq w^*$. If $w \in V[s_v^i, d_{s_v^i}]$, then

$$\begin{aligned} k_w^4 &= \sum_{u \in \mathcal{V}} r_u \cdot x_{uw}^4 \\ &= \sum_{u \in V[s_v^i, d_{s_v^i}]} r_u \cdot x_{uw}^1 + \sum_{u \in V[v,i-1]} r_u \cdot x_{uw}^2 + \sum_{u \notin V[v,i]} r_u \cdot x_{uw}^4 \\ &= \sum_{u \in V[s_v^i, d_{s_v^i}]} r_u \cdot x_{uw}^1 + 0 + 0 \\ &= \sum_{u \in V[s_v^i, d_{s_v^i}]} r_u \cdot x_{uw}^1 + \sum_{u \in V[v,i-1]} r_u \cdot x_{uw}^1 + \sum_{u \notin V[v,i]} r_u \cdot x_{uw}^1 \\ &= \sum_{u \in \mathcal{V}} r_u \cdot x_{uw}^1 = k_w^1 \end{aligned}$$

where the third and fourth equality follow from the contiguity property and the definition of x^4. For $w \in V[v, i-1]$ and $w \notin V[v,i]$ similar results can be obtained.

ad (2.11) If $w \notin V[s_v^i, d_{s_v^i}]$ then $x_{s_v^i w}^4 = x_{s_v^i w}^1 = 0$. If $w \notin V[v,i]$ then $x_{vw}^4 = x_{vw}^2 = 0$.

ad (2.12) It holds that

$$\begin{aligned} \sum_{u \notin V[s_v^i, d_{s_v^i}], w \in V[s_v^i, d_{s_v^i}]} r_u \cdot x_{uw}^4 &= \sum_{u \in V[v,i-1], w \in V[s_v^i, d_{s_v^i}]} r_u \cdot x_{uw}^2 \\ &\quad + \sum_{u \notin V[v,i], w \in V[s_v^i, d_{s_v^i}]} r_u \cdot x_{uw}^4 \\ &= \sum_{u \in V[v,i-1]} r_u \cdot x_{uw^*}^2 + \sum_{u \notin V[v,i]} r_u \cdot x_{uw^*}^4 \\ &= \alpha + \sum_{u \notin V[v,i], u' \in V[v,i]} r_u \cdot x_{uu'}^3 = \alpha + s \end{aligned}$$

Moreover,

$$\sum_{u \notin V[v,i], w \in V[v,i]} r_u \cdot x_{uw}^4 = \sum_{u \notin V[v,i]} r_u \cdot x_{uw^*}^4 = \sum_{u \notin V[v,i], u' \in V[v,i]} r_u \cdot x_{uu'}^3 = s$$

ad (2.15) If $w \in V[v, i-1]$ then $x_{vw}^4 = x_{vw}^2 = 0$.

ad (2.16) $\sum_{u \in V[v,i-1], w \notin V[v,i-1]} r_u \cdot x_{uw}^4 = \sum_{u \in V[v,i-1], w \notin V[v,i-1]} r_u \cdot x_{uw}^2 = r$.

This completes the feasibility part of the proof. In order to prove equivalence of the costs, note that we have shown in the above that $k_w^4 = k_w^j$ with $j = 1, 2, 3$ depending on the location of w. Next we show that for all $e \in \mathcal{E}$: $\ell_e^4 = \ell_e^j$, where $j = 1, 2, 3$ depending on whether $e \in E[s_v^i, d_{s_v^i}] \cup \{s_v^i\}$, $e \in E[v, i-1]$ or $e \notin E[v,i]$. Consider the case where $e \in E[s_v^i, d_{s_v^i}] \cup \{s_v^i\}$. Note that, if $u, w \in \mathcal{V}$ are such that $e \in E(u,w)$ and $u \notin V[s_v^i, d_{s_v^i}]$, then $w \in V[s_v^i, d_{s_v^i}]$. Hence,

$$\begin{aligned} \ell_e^4 &= \sum_{\substack{u,w \in \mathcal{V}: \\ e \in E(u,w)}} r_u \cdot x_{uw}^4 = \sum_{\substack{u \in V[s_v^i, d_{s_v^i}], w \in \mathcal{V}: \\ e \in E(u,w)}} r_u \cdot x_{uw}^1 \\ &\quad + \sum_{\substack{u \in V[v,i-1], w \in V[s_v^i, d_{s_v^i}]: \\ e \in E(u,w)}} r_u \cdot x_{uw}^2 + \sum_{\substack{u \notin V[v,i], w \in V[s_v^i, d_{s_v^i}]: \\ e \in E(u,w)}} r_u \cdot x_{uw}^4 \\ &= \sum_{\substack{u \in V[s_v^i, d_{s_v^i}], w \in \mathcal{V}: \\ e \in E(u,w)}} r_u \cdot x_{uw}^1 \end{aligned}$$

$$
\begin{aligned}
&+ \left[\sum_{u\in V[v,i-1]} r_u \cdot x^2_{uw^*} + \sum_{u\notin V[v,i],u'\in V[v,i]} r_u \cdot x^3_{uu'}\right] \cdot 1_{[e\in E(v,w^*)]} \\
&= \sum_{\substack{u\in V[s^i_v,d_{s^i_v}],w\in\mathcal{V}:\\ e\in E(u,w)}} r_u \cdot x^1_{uw} + [\alpha+s]\cdot 1_{[e\in E(v,w^*)]} \\
&= \sum_{\substack{u\in V[s^i_v,d_{s^i_v}],w\in\mathcal{V}:\\ e\in E(u,w)}} r_u \cdot x^1_{uw} + \left[\sum_{u\notin V[s^i_v,d_{s^i_v}]} r_u \cdot x^1_{uw^*}\right] \cdot 1_{[e\in E(v,w^*)]} \\
&= \sum_{\substack{u\in V[s^i_v,d_{s^i_v}],w\in\mathcal{V}:\\ e\in E(u,w)}} r_u \cdot x^1_{uw} + \sum_{\substack{u\notin V[s^i_v,d_{s^i_v}],w\in\mathcal{V}:\\ e\in E(u,w)}} r_u \cdot x^1_{uw} = \ell^1_e
\end{aligned}
$$

In case $e \in E[v,i-1]$ or $e \notin E[v,i]$ similar results hold.

As a consequence, the first two cost statements follow trivially. Furthermore,

$$C(x^4,k^4,\ell^4 \mid T[v,i]) = C(x^1,k^1,\ell^1 \mid T[s^i_v,d_{s^i_v}]) + C(x^2,k^2,\ell^2 \mid T[v,i-1]) + L_{s^i_v}(\ell^4_{s^i_v})$$

Moreover, from ad (2.12) it follows that $\ell^4_{s^i_v} = s+\alpha$. This completes the proof. ■

2.3.5 Relations between Family Members

Finally, we derive the recursive relations on which our dynamic programming algorithm is based. The last proof is omitted, for reasons of similarity.

Proposition 2.3.1 *Consider (v,i) with $v \in \mathcal{V}$ and $i = 0$. If $g(v,i,s) < \infty$ then*

$$g(v,i,s) = K_v(r_v + s) \tag{2.18}$$

Proof. Since $V[v,0] = \{v\}$, it follows from (2.11), (2.3) and (2.8) that $x_{vv} = 1$. Furthermore,

$$
\begin{aligned}
k_v = \textstyle\sum_{u\in\mathcal{V}} r_u \cdot x_{uv} &= \textstyle\sum_{u\in V[v,i]} r_u \cdot x_{uv} + \sum_{u\notin V[v,i]} r_u \cdot x_{uv} \\
&= r_v \cdot x_{vv} + \textstyle\sum_{u\notin V[v,i],w\in V[v,i]} r_u \cdot x_{uw} = r_v + s
\end{aligned}
$$

The objective function in (2.10) thus amounts to $C(x,k,\ell \mid T[v,i]) = K_v(r+s)$. ■

Proposition 2.3.2 *Consider (v,i) with $v \in \mathcal{V}$ and $i = 0$. If $h(v,i,r) < \infty$ then $v \neq 0$, $r = r_v$, $r \le B - \min_{u\notin V[v,i]\,:\,\{u,v\}\in\mathcal{E}} \{r_u\}$ and*

$$h(v,i,r) = K_v(0) \tag{2.19}$$

Proof. From (2.2) and (2.15) it follows directly that $v \neq 0$. Moreover, $V[v,0] = \{v\}$ and (2.16) imply that $r = r_v$, and since the load r must home on a node w outside $T[v,i]$, by contiguity the first node on the path from v to w homes on w. Hence, $r \leq B - \min_{u \notin V[v,i]:\{u,v\} \in \mathcal{E}} \{r_u\}$. Finally, the objective function in (2.14) amounts to $C(x,k,\ell \mid T[v,i]) = K_v(0)$. ■

Proposition 2.3.3 *Consider (v,i) with $v \in \mathcal{V}$ and $1 \leq i \leq d_v$. Define*

$$A_\alpha = g(s_v^i, d_{s_v^i}, s+\alpha) + h(v,i-1,\alpha) + L_{s_v^i}(s+\alpha) \quad (r_v \leq \alpha \leq B - s - r_{s_v^i}) \tag{2.20}$$

$$B_\alpha = h(s_v^i, d_{s_v^i}, \alpha) + g(v,i-1,s+\alpha) + L_{s_v^i}(\alpha) \quad (r_{s_v^i} \leq \alpha \leq B - s - r_v) \tag{2.21}$$

$$C = g(s_v^i, d_{s_v^i}, 0) + g(v,i-1,s) + L_{s_v^i}(0) \tag{2.22}$$

$$D = \min\left[\min_{r_v \leq \alpha \leq B-s-r_{s_v^i}} \{A_\alpha\}, \min_{r_{s_v^i} \leq \alpha \leq r-r_v} \{B_\alpha\}, C\right] \tag{2.23}$$

Then $g(v,i,s) \geq D$. Moreover, if $g(v,i,s) < \infty$ then $g(v,i,s) = D$.

Proof. Let $(x,k,\ell) \in \mathcal{F}$ be an optimal solution for $g(v,i,s)$ (if such a solution does not exist then $g(v,i,s) = \infty \geq D$). By definition of $g(v,i,s)$, v homes in $T[v,i]$. If v homes in $T[s_v^i, d_{s_v^i}]$ then Lemma 2.3.5 implies for $\ell_{s_v^i} = s + \alpha$ that $r_v \leq \alpha \leq B - s - r_{s_v^i}$ and that (x,k,ℓ) is feasible for both $g(s_v^i, d_{s_v^i}, s+\alpha)$ and $h(v,i-1,\alpha)$. Furthermore,

$$\begin{aligned} g(v,i,s) &= C(x,k,\ell \mid T[v,i]) \\ &= C(x,k,\ell \mid T[s_v^i, d_{s_v^i}]) + C(x,k,\ell \mid T[v,i-1]) + L_{s_v^i}(s+\alpha) \geq A_\alpha \end{aligned}$$

For both v and s_v^i homing in $T[v,i-1]$ or for v homing in $T[v,i-1]$ and s_v^i homing in $T[s_v^i, d_{s_v^i}]$ it can be proven in an analogous manner that $g(v,i,s) \geq B_\alpha$ and $g(v,i,s) \geq C$, respectively. This establishes the fact that $g(v,i,s) \geq D$.

In order to prove the second part of the proposition, let $g(v,i,s) < \infty$. From the aforementioned result it follows that $D < \infty$. Suppose $D = A_{\alpha^*}$ for some α^* with $r_v \leq \alpha^* \leq B - s - r_{s_v^i}$. Let (x^1,k^1,ℓ^1) be an optimal solution to $g(s_v^i, d_{s_v^i}, s+\alpha^*)$, (x^2,k^2,ℓ^2) be an optimal solution to $h(v,i-1,\alpha^*)$ (the existence of both solutions is implied by the fact that $D = A_{\alpha^*} < \infty$), and (x^3,k^3,ℓ^3) be a feasible solution to $g(v,i,s)$ (for which the existence is guaranteed by $g(v,i,s) < \infty$). Applying Lemma 2.3.10 yields a feasible solution (x^4,k^4,ℓ^4), which is simultaneously optimal for $g(s_v^i, d_{s_v^i}, s+\alpha^*)$ and $h(v,i-1,\alpha^*)$, and feasible for $g(v,i,s)$. Consequently,

$$\begin{aligned} g(v,i,s) &\leq C(x^4,k^4,\ell^4 \mid T[v,i]) \\ &= C(x^1,k^1,\ell^1 \mid T[s_v^i, d_{s_v^i}]) + C(x^2,k^2,\ell^2 \mid T[v,i-1]) + L_{s_v^i}(s+\alpha^*) \\ &= g(s_v^i, d_{s_v^i}, s+\alpha^*) + h(v,i-1,\alpha^*) + L_{s_v^i}(s+\alpha^*) = A_{\alpha^*} = D \end{aligned}$$

If $D = B_{\alpha^*}$ for some α^* with $r_{s_v^i} \leq \alpha^* \leq r - r_v$, or $D = C$, it can be proven in a similar manner that $g(v,i,s) \leq D$. This establishes the result. ■

Proposition 2.3.4 *Consider* (v,i) *with* $v \in \mathcal{V}$ *and* $1 \le i \le d_v$. *Define*

$$E_\alpha = h(s_v^i, d_{s_v^i}, \alpha) + h(v, i-1, r-\alpha) + L_{s_v^i}(\alpha) \quad (r_{s_v^i} \le \alpha \le r - r_v) \tag{2.24}$$

$$F = g(s_v^i, d_{s_v^i}, 0) + h(v, i-1, r) + L_{s_v^i}(0) \tag{2.25}$$

$$G = \min\left[\min_{r_{s_v^i} \le \alpha \le r - r_v}\{E_\alpha\}, F\right] \tag{2.26}$$

Then $h(v,i,r) \ge G$. *Moreover, if* $h(v,i,r) < \infty$ *then* $h(v,i,r) = G$.

2.4 An $\mathcal{O}(nB^2)$ Algorithm for LATNEP

The relationships between the subproblems derived in the preceding section give rise to the following dynamic programming algorithm. Recall that $g(v,i,s)$ is only defined for (v,i) with $v \in \mathcal{V}$, $0 \le i \le d_v$ and $0 \le s \le B - r_v$, whereas $h(v,i,r)$ is defined on the same (v,i) pairs (excluding $v = 0$, since a concentrator is always installed in the root) with $r_v \le r \le B$.

DYNAMIC PROGRAMMING ALGORITHM FOR LATNEP

forall (v,i,s) with $v \in \mathcal{V}$, $0 \le i \le d_v$ and $0 \le s \le B - r_v$ **do**
    $g(v,i,s) = \infty$; /* *initialization* g */
forall (v,i,r) with $v \in \mathcal{V}$, $0 \le i \le d_v$ and $r_v \le r \le B$ **do**
    $h(v,i,r) = \infty$; /* *initialization* h */
forall $v = n$ **downto** 0 **do begin**
    forall s with $0 \le s \le B - r_v$ **do**
        $g(v,0,s) = K_v(r_v + s)$;
    if $(v \ne 0)$ **then**
        $h(v,0,r_v) = K_v(0)$;
    forall $i = 1$ **to** d_v **do begin**
        forall s with $0 \le s \le B - r_v$ **do**
            $g(v,i,s) = D$ with D defined as in (2.20)–(2.23);
        forall r with $r_v \le r \le B$ **do**
            $h(v,i,r) = G$ with G defined as in (2.24)–(2.26);
    end;
end; optimal solution: $g(0,d_0,0)$

Unfortunately, the correctness of the algorithm does not immediately follow from the results in Section 2.3.5, since Proposition 2.3.3–2.3.4 do not exclude the possibility that $D < g(v,i,s) = \infty$ or $G < h(v,i,r) = \infty$. Indeed, this situation may occur, since we ignore constraint (2.12) and (2.16) during computations. In Theorem 2.4.1 we give a mathematical proof of the correctness of the algorithm. In the forthcoming analysis we will distinguish between the values of $g(v,i,s)$ and $h(v,i,r)$ as defined in Section 2.3 on the one hand, and the ones that are computed by the aforementioned algorithm on the other hand, by temporarily providing the latter with a superscript "c" (of "computed"). Roughly speaking, we prove that the computed values $g^c(v,i,s)$ and $h^c(v,i,r)$ may differ from the true values $g(v,i,s)$ and $h(v,i,r)$ only if these problems are infeasible, which is sufficient to prove the main result of this chapter (summarized in Theorem 2.4.2).

Theorem 2.4.1 *Consider (v,i) with $v \in \mathcal{V}$ and $0 \leq i \leq d_v$. Let r and s be such that $r_v \leq r \leq B$ and $0 \leq s \leq B - r_v$. Then the following statements hold*

(i). If $g(v,i,s) < \infty$ then $g^c(v,i,s) = g(v,i,s)$;

(ii). If $r \leq B - \min_{u \notin V[v,i]:\{u,v\} \in \mathcal{E}} \{r_u\}$ then $h^c(v,i,r) = h(v,i,r)$;

Proof. First consider $i = 0$. If $g(v,i,s) < \infty$ then $g^c(v,i,s) = g(v,i,s)$ follows from (2.18) and the definition of $g^c(v,i,s)$ in the algorithm. If $r \leq B - \min_{u \notin V[v,i]:\{u,v\} \in \mathcal{E}} \{r_u\}$ then Proposition 2.3.2 states that $h(v,i,r) = K_v(0)$ if $r = r_v$, $v \neq 0$, and $h(v,i,r) = \infty$ if $r \neq r_v$ or $v = 0$. Hence, $h^c(v,i,r) = h(v,i,r)$ follows from (2.19) and the definition of $h^c(v,i,r)$ in the algorithm.

To complete the proof we use induction on the pairs (v,i), $i > 0$ in the order as described by the algorithm. If $g(v,i,s) < \infty$ then by Proposition 2.3.3 we have $g(v,i,s) = D$. If the minimum in (2.23) is attained for $A_{\tilde{\alpha}}$ for some $\tilde{\alpha}$ with $r_v \leq \tilde{\alpha} \leq B - s - r_{s_v^i}$, then

$$\begin{aligned} g(v,i,s) &= g(s_v^i, d_{s_v^i}, s + \tilde{\alpha}) + h(v,i-1,\tilde{\alpha}) + L_{s_v^i}(s+\tilde{\alpha}) \\ &= g^c(s_v^i, d_{s_v^i}, s + \tilde{\alpha}) + h^c(v,i-1,\tilde{\alpha}) + L_{s_v^i}(s+\tilde{\alpha}) \geq g^c(v,i,s) \end{aligned}$$

where the second equality follows from the induction hypothesis, since $g(s_v^i, d_{s_v^i}, s+\tilde{\alpha}) < \infty$ and $\tilde{\alpha} \leq B - s - r_{s_v^i} \leq B - r_{s_v^i} \leq B - \min_{u \notin V[v,i]:\{u,v\} \in \mathcal{E}} \{r_u\}$. If the minimum in (2.23) is attained for $B_{\tilde{\alpha}}$ for some $\tilde{\alpha}$ or for C, then $g(v,i,s) \geq g^c(v,i,s)$ follows similarly. Next we will prove the reverse inequality.

The aforementioned yields $g^c(v,i,s) \leq g(v,i,s) < \infty$. If the minimum for $g^c(v,i,s)$ is attained by (2.20), then

$$\begin{aligned} g^c(v,i,s) &= g^c(s_v^i, d_{s_v^i}, s + \tilde{\alpha}) + h^c(v,i-1,\tilde{\alpha}) + L_{s_v^i}(s+\tilde{\alpha}) \\ &= g(s_v^i, d_{s_v^i}, s + \tilde{\alpha}) + h(v,i-1,\tilde{\alpha}) + L_{s_v^i}(s+\tilde{\alpha}) \geq g(v,i,s) \end{aligned}$$

where the second equality is explained as follows. First, since the minimum is attained by (2.20), $g^c(s_v^i, d_{s_v^i}, s + \tilde{\alpha})$ is actually computed. From the algorithm it then follows that

$s + \tilde{\alpha} \leq B - r_{s_v^i}$ which implies that $\tilde{\alpha} \leq B - s - r_{s_v^i} \leq B - \min_{u \notin V[v,i-1]:\{u,v\} \in \mathcal{E}} \{r_u\}$. By the induction hypothesis we can therefore conclude that $h^c(v, i-1, \tilde{\alpha}) = h(v, i-1, \tilde{\alpha})$. Secondly, since $g(v, i, s) < \infty$, $h(v, i-1, \tilde{\alpha}) < \infty$, $s \leq B - \tilde{\alpha} - r_{s_v^i}$ and $\tilde{\alpha} \geq r_v$ we can construct a feasible solution for $g(s_v^i, d_{s_v^i}, s + \tilde{\alpha})$ using feasible solutions of $g(v, i, s)$ and $h(v, i-1, \tilde{\alpha})$ (see Flippo et al.[33] for the strict mathematical construction of this solution). From the induction hypothesis it follows that $g^c(s_v^i, d_{s_v^i}, s + \tilde{\alpha}) = g(s_v^i, d_{s_v^i}, s + \tilde{\alpha})$, which justifies the second equality.

If the minimum for $g^c(v, i, s)$ is attained by (2.21) for some $\tilde{\alpha}$ or by (2.22), then $g^c(v, i, s) \geq g(v, i, s)$ follows similarly. As a result, $g(v, i, s) < \infty$ implies $g^c(v, i, s) = g(v, i, s)$. For $h(v, i, r)$, the result follows in a similar way. ■

Theorem 2.4.2 *Suppose $K_v(k_v)$ can be computed in $\mathcal{O}(m)$ time for every $k_v \in [r_v, B] \cap \mathbb{N}$ and $v \in \mathcal{V}$, and $L_e(\ell_e)$ can be computed in $\mathcal{O}(p)$ time for every $\ell_e \in [0, B] \cap \mathbb{N}$ and $e \in \mathcal{E}$, where m and p are parameters depending on problem size. Furthermore, let each of these computations require $\mathcal{O}(nB)$ storage space. Then the aforementioned dynamic programming algorithm finds an optimal solution in $\mathcal{O}(n(m+p)B + nB^2)$ time and $\mathcal{O}(nB)$ storage space. Under mild conditions on the cost structure (which are satisfied in the real-life applications of Balakrishnan, Magnanti and Wong [9] and Cho and Shaw [23]), it follows that $\mathcal{O}(m) = \mathcal{O}(B)$ and $\mathcal{O}(p) = \mathcal{O}(1)$, implying an overall time complexity of $\mathcal{O}(nB^2)$.*

Proof. Correctness follows directly from $g^c(0, d_0, 0) = g(0, d_0, 0)$ (cf. Theorem 2.4.1). As for time and space complexity, all coefficients $K_v(k_v)$ can be calculated in $\mathcal{O}(nmB)$ and all coefficients $L_e(\ell_e)$ in $\mathcal{O}(npB)$. Storage requirements for these coefficients is $\mathcal{O}(nB)$. Since the number of (v, i)-pairs we consider is $\mathcal{O}(n)$, it follows that all remaining computations can be done in $\mathcal{O}(nB^2)$. Obviously, the storage requirement for all g and h coefficients equals $\mathcal{O}(nB)$.

In practical situations it is reasonable to assume that $\mathcal{O}(m) = \mathcal{O}(B)$. For instance, in the cost structure described by Cho and Shaw [23] (cf. Section 2.2), the variable concentrator costs $\hat{c}_v$ do not depend on the concentrator type t. So, if two concentrators t_1 and t_2 have the same capacity $\hat{b}^{t_1} = \hat{b}^{t_2}$, and if $\hat{F}_v^{t_1} \leq \hat{F}_v^{t_2}$, then t_1 will never be more expensive to use than t_2. In other words, for every node $v \in \mathcal{V}$ there will be at most one non-dominated concentrator per load figure k_v, implying that $\min_t\{\hat{F}_v^t + \hat{c}_v k_v \mid \hat{b}^t \geq k_v\}$ can be computed in $\mathcal{O}(m) = \mathcal{O}(B)$ time. Similarly, in Balakrishnan, Magnanti and Wong [9], the concentrator cost structure is represented by a piecewise-linear, concave function, with breakpoints occurring only at integer-valued arguments. Since such a function is described as the point-wise minimum of at most B affine functions, it follows again that $\mathcal{O}(m) = \mathcal{O}(B)$. ■

Now we have established the correctness of the procedure, we will drop the superscript "c" henceforth. Note that the algorithm in its current form only determines the optimal solution *value*, rather then an optimal *solution*. However, from Propositions 2.3.1–2.3.4 it follows that an optimal solution can be recovered in the traditional way by keeping track of the "argmin" (instead of just the "min") in the evaluations of g and h coefficients, and by constructing the solution afterwards using backward recursion.

2.5 Computational Results

To assess the practical feasibility of the dynamic programming algorithm, we have implemented the algorithm in the programming language C++ on a DEC 2100 A500MP workstation with 128Mb internal memory. We tested the algorithm both on generated instances resembling Cho and Shaw's [23] and on the real-life 41 node instance of Balakrishnan, Magnanti and Wong [9].

2.5.1 Generated Problem Instances

All generated problem instances have the same cost structure as the one proposed by Cho and Shaw [23] (cf. Section 2.2). The first set of instances we consider has been made publicly available by Cho and Shaw. The number of nodes in the tree varies from 5 to 30, whereas the maximum concentrator capacity is in the range of 41 to 951. The results in Table 2.1 indicate that our algorithm is competitive to Cho and Shaw's algorithm; it is 43 times as fast on average over all 11 problems, and even up to 76 times as fast on average over the 5 largest problems. (Note that this may partly be due to differences in hardware; Cho and Shaw used a SUN SPARC 1000 workstation.)

Table 2.1: Computational results for the instances generated by Cho and Shaw.

problem	n	m	B	value	CPU sec.
exp5a	5	1	41	559	0.017
exp9a	9	1	51	1305	0.020
exp9b	9	1	70	559	0.020
exp9c	9	1	340	25847	0.023
exp15a	15	3	451	55566	0.028
exp15b	15	3	474	65693	0.039
exp20a	20	1	140	23382	0.029
exp20b	20	3	443	62859	0.038
exp20c	20	3	951	162936	0.059
exp30a	30	1	340	93238	0.044
exp30b	30	3	451	86717	0.077

We also tested 10 larger instances of this type using the same generator as Cho and Shaw (available on Shaw's Web site), with the number of nodes varying from 50 to 200, and with the maximum concentrator capacity varying from 100 to 1500. Our results for these instances are listed in Table 2.2, which indicate that much larger instances can still be solved efficiently: instances with small concentrator capacities are solved within a second, whereas the running time for larger concentrator capacities is within minutes.

All of the above instances consisted of trees with an unbalanced structure as is illustrated in Figure 2.1. Due to this structure, the number of coefficients r for which $h(v,i,r)$ is

Table 2.2: Computational results for instances with 50 to 200 nodes.

problem	n	m	B	value	CPU sec.
`lanep50a`	50	3	185	247768	0.058
`lanep50b`	50	3	392	197222	0.363
`lanep50c`	50	3	928	127577	3.311
`lanep100a`	100	3	190	632303	0.181
`lanep100b`	100	3	280	443484	0.539
`lanep100c`	100	3	772	342672	10.045
`lanep200a`	200	3	137	1453570	0.193
`lanep200b`	200	3	192	1417313	0.450
`lanep200c`	200	3	680	795365	13.992
`lanep200d`	200	3	1424	574739	80.648

feasible for the (v, i)-pairs with $v \in \{0, 16, 31, 35, 45\}$ is very large, which makes this type of instances relatively hard to solve. Therefore, we slightly modified Cho and Shaw's problem generator so as to obtain more balanced trees. We generated trees consisting of 25 up to 1000 nodes, and concentrator capacities ranging from 3000 to over 10000. For each of the problem sizes, we generated five instances with 3 concentrator types. The best, average and worst CPU times are reported in Table 2.3. The results indicate that small problem instances can be solved in a second, whereas the larger instances can be solved in a minute. An implementation of the algorithm as well as the modified generator with all of the aforementioned instances are publicly available on World Wide Web or by e-mail to the author.

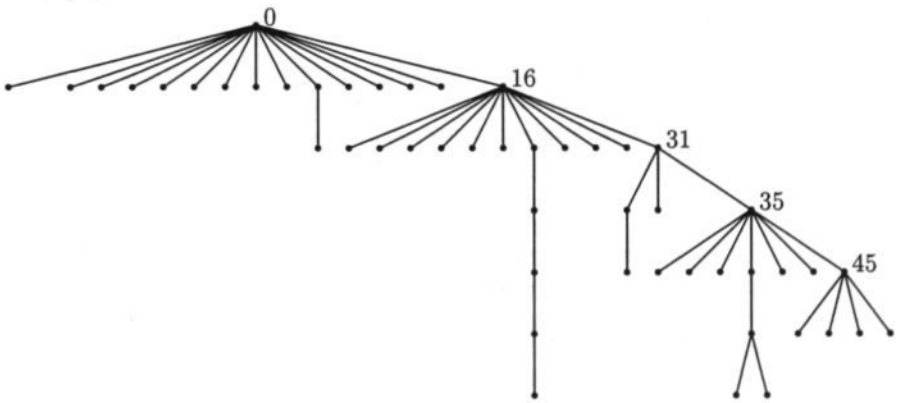


Figure 2.1: The Tree of Problem `lanep50a`

Table 2.3: Computational results for balanced instances with 25 to 1000 nodes.

n	min B	max B	CPU sec.		
			best	average	worst
25	4408	4869	0.339	0.507	0.665
50	4169	4763	0.697	0.934	1.248
100	3761	5360	1.161	1.795	2.602
200	7708	10218	31.535	71.680	127.619
500	3800	4762	8.335	11.407	18.209
1000	2907	3597	9.873	13.423	16.184

2.5.2 Real-Life Problem Instances

Balakrishnan, Magnanti and Wong [9] propose a solution method which incorporates valid inequalities in a dynamic program to solve the uncapacitated version of the problem (i.e. the design problem). For three realistic networks they embed their dynamic program in a Lagrangian relaxation scheme to obtain solutions for the capacitated version (i.e. LATNEP) within 1.2–7.0% of optimality and 15 minutes of running time. Unfortunately, these instances are not publicly available. However, in their article all demands and existing cable capacities for the largest (41) node instance are given. Since our algorithm is basically independent of the cost structure (assuming that the concentrator and cable cost can be calculated in $\mathcal{O}(B)$ and $\mathcal{O}(1)$ time), we tested our algorithm on the 41 node instance with a Cho and Shaw cost structure. In Table 2.4 the computation times for several values of B are given; the largest B is hereby set equal to the sum of all demands. We also tested this instance on a Pentium 166 Mhz personal computer with 16 Mb internal memory, on which we needed 105 seconds to compute the optimal solution (with $B = 43,212$).

Table 2.4: Computational results for the Balakrishnan, Magnanti and Wong instance with 41 nodes.

B	CPU sec.
10000	2.875
20000	13.655
30000	33.993
43212	44.600

Chapter 3

A Dynamic Programming Algorithm for the ATM Tree Network Installation Problem

3.1 Introduction

Modern telecommunication networks are capable of processing multiple telecommunication services on a single physical network. These so-called broadband networks usually consist of several hierarchical network layers. At the top layer (often referred to as the Backbone), large capacity nodes serve large geographical areas. These areas are decomposed into smaller regions, each of which is served by nodes located in lower layers of the network. In the Backbone, connectivity between nodes is usually fairly high. This is due both to high traffic requirements, which make the installation of direct links between nodes economically attractive, and to reliability considerations, which coerce that in case of a calamity in the network (the breakdown of a node or a link), alternative routing between pairs of nodes must be available. In lower levels of the network both the intensity of traffic and the need for reliability decrease, and the structure commonly used here is thus a tree.

In order to send information of different types of services over the same broadband network, a structured protocol is required. Nowadays ATM (Asynchronous Transfer Mode) is widely accepted as the standard protocol for (future) broadband networks. In the ATM protocol, a message that needs to be sent from its source to its destination is decomposed into small units of information which are stored in ATM cells. In addition to a part of the message, an ATM cell also contains certain routing information, which enables the cell to be routed through the network from source to destination. At the destination node, the different units are combined to reconstruct the original message. To implement the ATM protocol on the telecommunication network, certain hardware devices have to be installed, connections between these devices have to be made, and customers must be connected to these devices. As such, an ATM network topology is built upon an existing network struc-

ture. The problem of designing and maintaining telecommunication networks typically involves simultaneous decisions regarding the location of hardware devices, the capacity of these devices, the capacity installation on connections between devices, and the routing of messages over these connections. Since these problems are by far too complex to be handled at once, they are often decomposed.

In this chapter, based on Van de Leensel, Flippo and Koster [75], we discuss the ATM Tree Network Installation Problem (denoted ATNIP in the sequel) on a given tree, which consists of all the nodes in the area being served by the same Backbone node. The root of the tree is thus a node in the Backbone, in which an ATM hardware device is installed by default. Traffic demand between pairs of nodes in the tree are also given (these are called commodities). To enable communication between the endpoints of a commodity, capacity has to be installed on the edges of a *walk* in the tree between the nodes involved. Note that a walk is defined here as a path in which edge repetition is allowed. The reason for this edge repetition will become clear in Section 3.2. By installing a hardware device (called an ATM cross connect) in a node, ATM signals can be combined such that the required capacity on the edges can be reduced. The key issue in ATNIP is thus to find a trade-off between costs of capacity installation on edges on the one hand, and costs of installing ATM cross connects in nodes on the other hand, such that all demand is met and total costs are minimized.

In the sequel, ATNIP is shown to be NP-hard, as it contains SUBSET SUM as a special case. We state a pseudo-polynomial time dynamic programming algorithm which runs in $\mathcal{O}(nB^4)$ and requires $\mathcal{O}(nB^2)$ storage space. Here, n refers to the number of nodes in the tree, and B to an upper bound on the capacity of an ATM cross connect. Although the algorithm can easily be applied when demand is non-symmetric, for ease of exposition we will assume throughout this paper that demand is indeed symmetric. Computational experiments indicate that our algorithm runs efficiently on real-life problem instances, made available to us by KPN Research, Leidschendam, The Netherlands.

The remainder of this chapter is organized as follows. In Section 3.2 we give a detailed description of different aspects of the functionality of ATM networks and its components, for as far as they are relevant for the problem at hand. The notation used throughout this paper and a mathematical formulation of the problem are stated in Section 3.3. In Section 3.4 two families of subproblems are defined, and the relations between these subproblems are discussed. The dynamic programming algorithm and its use as an ATM network planning tool are the subject of Section 3.5. Computational results conclude the paper in Section 3.6.

3.2 ATM functionality

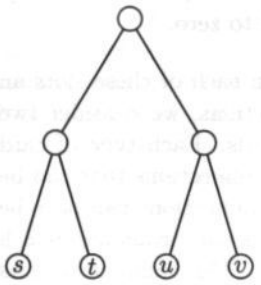
In this section we give a detailed description of an ATM tree network and the hardware devices it may contain. We confine ourselves to those functionalities which are important for the problem at hand. An ATM tree network consists of ATM cross connects (to be referred to as switches henceforth) that can be installed in the nodes of the tree, and cables

between these switches (usually referred to as connections). If a connection is installed between two switches, it actually requires capacity installation on each edge of the (unique) path between the end points of the connection. We consider two types of connections, viz. 34Mb/s and 155Mb/s connections (of course the model can be extended to incorporate other types of connections as well). The capacity of a connection is bi-directional, i.e. a 34Mb/s connection can accommodate 34Mb/s in both directions simultaneously. Connections can be used to transport information between demand nodes (nodes in the underlying tree structure) and switches on the one hand, and between switches themselves on the other. We assume that all nodes in the tree are demand nodes, that is each node in the tree is the source (and by symmetry of demand also the destination) of a commodity. If necessary, the demand of certain commodities can be set equal to zero.

A switch contains a given number of slots (cf. Figure 3.1(c)). In each of these slots an interface card of a certain type can be installed. As for connections, we consider two types of interface cards, viz. 34Mb/s and 155Mb/s interface cards. Each type of card has a number of input/output gates which equals the number of connections that can be connected to the interface card. Obviously, 34Mb/s (155Mb/s) connections can only be connected to 34Mb/s (155Mb/s) cards. Different types of switches are available, which for our purpose only differ in the number of slots they provide. This number of slots determines a switch's capacity. Apart from slots a switch also contains a routing table. An ATM cell that enters a switch via an input/output gate of one of the interface cards installed, is passed through the routing table, which evaluates the routing information in the cell to determine on which connection the cell should leave the switch. This gives the switch the capability to separate two (or more) commodities which enter the switch via the same connection by sending them out on different connections. Of course the reverse, i.e. bundling of commodities, is also possible.

The overall ATM tree network should now be constructed in such a way that each commodity is routed from its source node to its destination node. This route may pass several ATM cross connects on its way. A connection must be installed on this route in any of three cases : on the unique path between each pair of consecutive ATM cross connects on this route, between the source of a commodity and its nearest ATM cross connect on the route, and between the destination of the commodity and its nearest ATM cross connect. Furthermore, a sufficient number of interface cards should be installed in the ATM cross connects in order to connect the connections to the ATM cross connects. Since the complete route from source node to destination node for a commodity is thus a concatenation of connections, the route is often called a virtual path in telecommunication literature. However, since the concatenation of paths (which itself do not allow for edge repetition) implies that on the complete route itself edge repetition might occur when a specific edge occurs in more than one of these paths, from a graph theoretical point of view it should really be referred to as a walk (which by definition allows for edge repetition). To explain the above in more detail, consider the illustration in Figure 3.1.

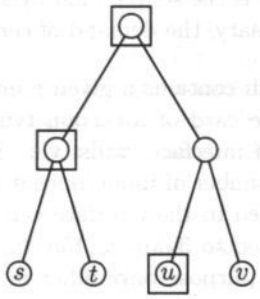
In order to enable communication between demand nodes s, t, u and v, all of these nodes must be connected to a switch. The switch to which a demand node is connected is called its "homing node" (or "homing switch"). Suppose that a switch is installed in the parent node of node s and t, serving as a homing node for nodes s and t, that a switch is



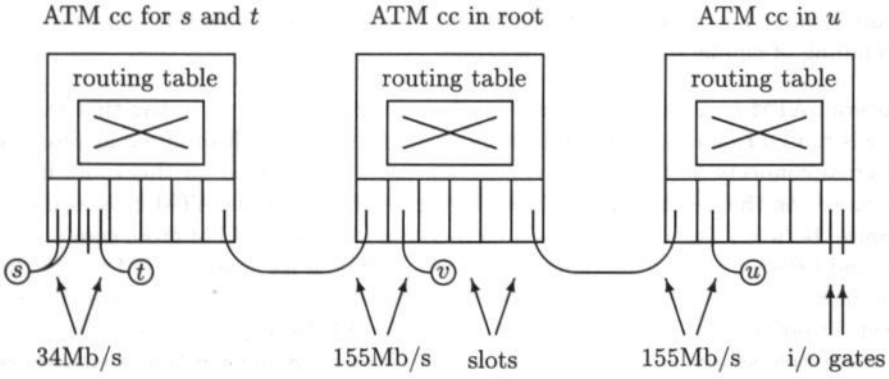
Demands in Mb/s :

	s	t	u	v
s	–	0	20	20
t	0	–	20	0
u	20	20	–	10
v	20	0	10	–

(a) problem instance



(b) switch installation



(c) detailed description

Figure 3.1: ATM Functionality: Example Instance

installed in u which is the homing node for node u, and that a final switch is installed in the root, which serves as the homing node for node v. This configuration is depicted in Figure 3.1(b), where squared boxes indicate the installation of a switch. If the origin and destination of a commodity have a different homing node, a virtual walk (as explained in the above) between the two homing nodes must be installed. In general, this walk may pass other switches. Figure 3.1(c) shows what type of connections may be added to enable the desired communication between nodes s, t, u and v (the proposed configuration is not claimed to be optimal in any sense, it just serves the explanation). Commodities (s, u) and (s, v) enter the homing switch of s on two 34Mb/s connections, and a single 34Mb/s connection is used for commodity (t, u). Next, the three commodities are bundled and transferred to the switch in the root on a single 155Mb/s connection. There the two commodities (s, u) and (t, u) are simply passed on via a 155Mb/s connection, towards the switch in node u. The commodity (s, v) is transported to node v via a 34Mb/s connection. Note that these connections pass two edges in the underlying graph, and therefore require capacity installation on both edges. The installed connections as described in the above also leave enough unused capacity to route the commodity (u, v). This commodity is routed from u to the switch in node u, then to the switch in the root, and then to node v. Note that to route the commodities from a demand node to its homing switch, and to separate/combine commodities at a demand node, some hardware device other than a switch must be installed at the demand node as well. In general, such hardware devices must be installed at every demand node. Therefore, these devices do not influence the design problem, and are left out of consideration. The reverse walks can be used for the reversed commodities. In the solution described in the above and depicted in Figure 3.1, edge repetition occurs, since the commodity (u, v) passes an edge in the underlying graph twice. Next, the reader may note that 34Mb/s (155Mb/s) interface cards have two (one) input/output gates. We will say more about the influence of the number of input/ouput gates in Section 3.3.

Network planners often impose certain restrictions on the layout of telecommunication networks. Some of these restrictions may be due to technical requirements, whereas others are considered to be economically sensible, or practical for operational convenience regarding maintenance and repair. For ATNIP, the following restrictions apply.

1. **Upward homing of demand nodes:** a demand node always homes on the first switch located on the unique path from the demand node to the root of the tree (both endpoints including);

2. **Upward homing of switches:** connections between two switches are only possible if one switch "homes" on the other, where a switch in node v (other than the root node) always homes on the first switch located on the path from (but excluding) v to the root. For example in Figure 3.1, the switch in node u is said to be homing on the switch in the root node, and the same holds for the switch located in the parent node of node s and t.

The reader may observe that the ATM network proposed in the example instance satisfies both routing restrictions.

Next, it needs to be stated that a commodity may not be split over different connections. This implies that 3 commodities, each with a demand of 20 Mb/s, cannot be routed on two 34Mb/s connections (although the total capacity seems to suffice), since it would require one of the commodities to be split over two connections. Hence, the 155Mb/s connection between the homing switch for node s and t and the switch in the root of the tree cannot be replaced by two 34Mb/s connections. Define J_u as the set of possible connections between a demand node u and its homing switch. For example, if the largest capacity switch contains 10 slots, and the number of input/ouput gates on a 34Mb/s and 155Mb/s interface card equals two and one, respectively, then J_u consists of twenty 34Mb/s connections and ten 155Mb/s connections. Furthermore, let D_u be the set of commodities which are to be sent from a demand node u to its homing switch (all in the same direction). Finally, define the following decision variables and parameters.

$$m_u^j = \begin{cases} 1 & \text{if connection } j \text{ is used} \\ 0 & \text{otherwise} \end{cases} \qquad (j \in J_u)$$

$$t_u^{ij} = \begin{cases} 1 & \text{if commodity } i \text{ is assigned to connection } j \\ 0 & \text{otherwise} \end{cases} \qquad (i \in D_u,\ j \in J_u)$$

$$\delta_u^i = \text{demand of commodity } i \qquad (i \in D_u)$$

$$cap_u^j = \text{capacity of connection } j \qquad (j \in J_u)$$

Then the collection of connections required to route the commodities of demand node u to its homing switch should satisfy the following restrictions:

$$\sum_{j \in J_u} t_u^{ij} = 1 \qquad \forall i \in D_u \qquad (3.1)$$

$$\sum_{i \in D_u} \delta_u^i \cdot t_u^{ij} \leq cap_u^j \cdot m_u^j \qquad \forall j \in J_u \qquad (3.2)$$

$$m_u^j \in \{0,1\},\ t_u^{ij} \in \{0,1\} \qquad \forall i \in D_u,\ \forall j \in J_u \qquad (3.3)$$

Likewise, for all commodities which must be routed from a switch to its homing switch, the required collection of connections involved should satisfy a similar set of constraints. This problem is referred to as the bin-packing problem of commodities on connections. We will return to this issue in Section 3.3, after we have given a mathematical formulation of the problem.

The costs of an ATM network structure stem from switches and connections. The following assumptions are made regarding to the costs:

- the costs of a switch are fully determined by the number of connections of each type connected to it, and by the node in which it is installed;
- the costs of a connection equal the sum of the costs on the edges on the connection (path);
- the costs of the switches are independent of the costs of the connections.

Note that these assumptions allow for very general costs structures, and encompass the typical fixed-charge cost functions observed in practice. Now all ingredients for the ATNIP are discussed in words, a formal description can be given.

3.3 Notation and Mathematical Formulation

$G(\mathcal{V},\mathcal{E})$	=	tree G, where $\mathcal{V}$ represents the node set and $\mathcal{E}$ the undirected edge set; the nodes are numbered in a depth first order with the root numbered 0; the edges are also numbered in a depth first order such that edge v is the edge between node v and its unique parent
d_v	=	the number of children of node v; the children are denoted by $s_v^1, \ldots, s_v^{d_v}$
$T[v,i]$	=	subtree induced by v, its first i children and all successors of these children
$V[v,i]$	=	set of nodes in subtree $T[v,i]$
$E[v,i]$	=	set of edges in subtree $T[v,i]$
$V(u,w)$	=	set of all nodes on the path from u to w, including both endpoints
$E(u,w)$	=	set of all edges on the path from u to w
X_u	=	set of possible homing nodes for a demand node u, i.e. $X_u = V(u,0)$
Y_u	=	set of possible homing nodes for a switch in node u, i.e. $Y_u = V(u,0)\backslash\{u\}$
T	=	set of available capacities (interface cards), e.g. $T = \{34, 155\}$ let $t = \|T\|$ denote the cardinality of T
$K_w(k_w)$	=	costs of a switch located in w if k_w represents the number of connections that are connected to this switch ($k_w \in \mathbb{N}^t$); $K_w(0)$ corresponds to the situation where no switch is installed in node w
$L_e(\ell_e)$	=	costs on an edge $e \in \mathcal{E}$ if ℓ_e represents the number of connections over edge e ($\ell_e \in \mathbb{N}^t$)
γ	=	number of input/output gates on interface cards ($\gamma \in \mathbb{N}^t$)
B	=	number of slots available in the largest capacity switch
D_u	=	set of commodities with source node u
P_u	=	set of commodities with its source node located in $T[u,d_u]$ and with destination node outside $T[u,d_u]$
J_u	=	set of available connections between demand node u and its homing switch; let $J_u = \cup_{\tau\in T} J_u^\tau$
H_u	=	set of available connections between a switch in u (given that a switch is installed there) and its homing switch; let $H_u = \cup_{\tau\in T} H_u^\tau$
δ_u^i	=	demand of commodity $i \in D_u$
cap_u^j	=	capacity of connection $j \in J_u$ or H_u

Note that $T[v,i] = T[v,i-1] \cup T[s_v^i, d_{s_v^i}] \cup \{s_v^i, v\}$. Reconsider Figure 1.5, then it holds that $T[4,2]$ can be decomposed into $T[4,1]$, $T[8,0]$ and edge $\{8,4\}$. This decomposition will be exploited frequently. Moreover, note that $P_u = \emptyset$ for $u = 0$, and define $H_u = \emptyset$, for $u = 0$. Next we introduce the following decision variables:

$$x_{uw} = \begin{cases} 1 & \text{if demand node } u \text{ is homing on a switch in node } w \\ 0 & \text{otherwise} \end{cases} \qquad (u \in \mathcal{V}, w \in X_u)$$

$$y_{uw} = \begin{cases} 1 & \text{if a switch in } u \text{ is homing on a switch in } w \\ 0 & \text{otherwise} \end{cases} \quad (u \in \mathcal{V} \setminus \{0\}, w \in Y_u)$$

$$m_u^j = \begin{cases} 1 & \text{if connection } j \text{ is used} \\ 0 & \text{otherwise} \end{cases} \quad (u \in \mathcal{V}, j \in J_u)$$

$$t_u^{ij} = \begin{cases} 1 & \text{if commodity } i \text{ is assigned to connection } j \\ 0 & \text{otherwise} \end{cases} \quad (u \in \mathcal{V}, i \in D_u, j \in J_u)$$

$$n_u^j = \begin{cases} 1 & \text{if connection } j \text{ is used} \\ 0 & \text{otherwise} \end{cases} \quad (u \in \mathcal{V}, j \in H_u)$$

$$z_u^{ij} = \begin{cases} 1 & \text{if commodity } i \text{ is assigned to connection } j \\ 0 & \text{otherwise} \end{cases} \quad (u \in \mathcal{V}, i \in P_u, j \in H_u)$$

$$k_w^\tau = \text{number of connections of type } \tau \text{ incident to } w \quad (w \in \mathcal{V}, \tau = 1, \ldots, t)$$

$$\ell_e^\tau = \text{number of connections of type } \tau \text{ on edge } e \quad (e \in \mathcal{E}, \tau = 1, \ldots, t)$$

The ATNIP is now given by the following mathematical formulation :

$$\min \sum_{w \in \mathcal{V}} K_w(k_w) + \sum_{e \in \mathcal{E}} L_e(\ell_e) \tag{3.4}$$

$$\text{s.t.} \sum_{w \in X_u} x_{uw} = 1 \qquad \forall u \in \mathcal{V} \tag{3.5}$$

$$x_{u'w} \geq x_{uw} \qquad \forall u, u', w \in \mathcal{V} : w \in Y_u, u' \in V(u,w) \setminus \{u\} \tag{3.6}$$

$$\sum_{w \in Y_u} y_{uw} = x_{uu} \qquad \forall u \in \mathcal{V} \setminus \{0\} \tag{3.7}$$

$$y_{uw} \leq x_{u'w} \qquad \forall u, u', w \in \mathcal{V} : w \in Y_u, u' \in V(u,w) \setminus \{u\} \tag{3.8}$$

$$\sum_{j \in J_u} t_u^{ij} = 1 \qquad \forall u \in \mathcal{V}, \forall i \in D_u \tag{3.9}$$

$$\sum_{i \in D_u} \delta_u^i \cdot t_u^{ij} \leq cap_u^j \cdot m_u^j \qquad \forall u \in \mathcal{V}, \forall j \in J_u \tag{3.10}$$

$$\sum_{j \in H_u} z_u^{ij} = x_{uu} \qquad \forall u \in \mathcal{V}, \forall i \in P_u \tag{3.11}$$

$$\sum_{i \in D_u} \delta_u^i \cdot z_u^{ij} \leq cap_u^j \cdot n_u^j \qquad \forall u \in \mathcal{V}, \forall j \in H_u \tag{3.12}$$

$$k_w^\tau = \sum_{u \in \mathcal{V}} \Big(\sum_{j \in J_u^\tau} m_u^j x_{uw} + \sum_{j \in H_u^\tau} n_u^j y_{uw} \Big) + \sum_{j \in H_w^\tau} n_w^j x_{ww} \qquad \forall w \in \mathcal{V}, \tau = 1, \ldots, t \tag{3.13}$$

$$\ell_e^\tau = \sum_{u,w \in \mathcal{V} : e \in E(u,w)} \Big(\sum_{j \in J_u^\tau} m_u^j x_{uw} + \sum_{j \in H_u^\tau} n_u^j y_{uw} \Big) \qquad \forall e \in \mathcal{E}, \tau = 1, \ldots, t \tag{3.14}$$

$$x_{uw} \in \{0,1\} \qquad \forall u \in \mathcal{V}, \forall w \in X_u \tag{3.15}$$

$$y_{uw} \in \{0,1\} \qquad \forall u \in \mathcal{V} \setminus \{0\}, \forall w \in Y_u \tag{3.16}$$

$$k_w,\ \ell_e \geq 0 \qquad \forall w \in \mathcal{V}, \forall e \in \mathcal{E} \tag{3.17}$$

$$m_u^j,\ n_u^j,\ t_u^{ij},\ z_u^{ij} \in \{0,1\} \qquad \forall u \in \mathcal{V}, \forall i \in D_u, P_u,\ \forall j \in J_u, H_u \tag{3.18}$$

The objective function in (3.4) defines the total costs for a solution (x, y, m, t, n, z, k, l) and

reflects the decomposability of the costs on switches installed in nodes and connections passing over edges. Constraint (3.5) implies that every demand node has exactly one homing node, and by the definition of X_u, it follows that a switch is installed in the root node. Constraint (3.6) enforces the upward homing condition of demand nodes, whereas constraint (3.7) and (3.8) guarantee the upward homing of switches. The bin-packing problem of commodities on connections is modeled by constraints (3.9) and (3.10) for a demand node and its homing switch, and by constraints (3.11) and (3.12) for commodities between a switch and its homing switch, respectively. Constraints (3.13) and (3.14) simply define the resulting number of connections connected to a switch in a node, and the number of connections which pass over an edge, respectively. The integrality and non-negativity constraints in (3.15)–(3.18) complete the formulation. Note that we have not listed a constraint which restricts the number of connections incident to a switch, which could have been formulated as

$$\sum_{\tau=1}^{t} \lceil k_w^\tau / \gamma^\tau \rceil \leq B \qquad \forall w \in \mathcal{V} \tag{3.19}$$

For ease of exposition, (3.19) will be stated as $k_w \ll B$ in the sequel, and should be interpreted as "the number of connections k_w fits on the largest capacity switch". Note that if a given number of connections k_w would require more slots than there are available in the largest capacity switch, this infeasibility can easily be incorporated in the objective function by setting the corresponding $K_w(k_w)$ equal to infinity. Hence, constraint (3.19) is not part of our model. In Section 3.5 however, the number B will be shown to be an important determinant for the running time of our algorithm.

The objective function (3.4) may encompass very general costs structures. For the typical fixed-charge cost structure arising in practice, the objective function can be specified as follows. Let θ refer to the different types of switches available, B_θ denote its capacity, and $\hat{F}_w(\theta)$ represent the fixed costs of installing a switch of type θ in node w, and μ_w^τ be the costs of an interface card of type τ in node w. Then

$$K_w(k_w) = \begin{cases} \min_{\theta: k_w \ll B_\theta} \{\hat{F}_w(\theta) + \sum_{\tau=1}^{t} \mu_w^\tau \cdot \lceil k_w^\tau / \gamma^\tau \rceil\} & k_w \neq 0 \\ 0 & k_w = 0 \end{cases} \tag{3.20}$$

with the minimum over an empty set defined as infinity. Throughout this chapter, we will assume that large capacity switches are available (since this is also the case in all real-life problem instances), which implies that a feasible solution always exists. Moreover, if λ_e^τ denotes the costs of installing a connection of type τ on an individual edge e, then

$$L_e(\ell_e) = \sum_{\tau=1}^{t} \lambda_e^\tau \cdot \ell_e^\tau \tag{3.21}$$

The reader may notice that for a given choice of the variables x_{uw}, y_{uw}, the remaining variables $t_u^{ij}, z_u^{ij}, m_u^j, n_u^j$ only interact with the previous mentioned variables via the objective function, since constraints (3.13) and (3.14) merely define the variables k_w and ℓ_e. For general objective functions (as well as parameter values), this interaction may of course

imply that the optimal values of the two classes of decision variables cannot be determined independently. Next, we derive conditions (which are fulfilled in the practical problem instances obtained from KPN Research, Leidschendam, The Netherlands), under which the number of connections used between a demand node and its homing switch (variables t_u^{ij}, m_u^j), as well as the number of connections used between a switch and its homing switch (variables z_u^{ij}, n_u^j), can be determined optimally beforehand, that is, independently of the choice of the remaining variables.

Lemma 3.3.1 *Consider ATNIP with objective function as defined by (3.4), (3.20) and (3.21), and with $t = 2$, i.e. two types of connections (interface cards), viz. 34Mb/s and 155Mb/s. If*

- *the interface card costs are positive and independent of its type, i.e. $\mu_w^\tau = \mu_w > 0$ for all τ and for all $w \in \mathcal{V}$*
- *the costs of installing two 34Mb/s connections on an edge are at least the costs of installing one 155Mb/s connection on an edge, and positive, i.e. $2 \cdot \lambda_e^{34} \geq \lambda_e^{155} > 0$, for all $e \in \mathcal{E}$*
- *the number of input/output gates on a 34Mb/s and 155Mb/s interface card is two and one, respectively, i.e. $\gamma^{34} = 2, \gamma^{155} = 1$*

then there exists an optimal solution (x, y, m, t, n, z, k, l) for ATNIP in which each demand node is connected to its homing switch using at most one 34Mb/s connection, i.e. $\sum_{j \in J_u^{34}} m_u^j \leq 1$, $\forall u \in \mathcal{V}$, and each switch is connected with its homing switch using at most one 34Mb/s connection, i.e. $\sum_{j \in H_u^{34}} n_u^j \leq 1$, $\forall u \in \mathcal{V} \setminus \{0\}$.

Proof. Suppose we are given a solution for ATNIP in which a demand node u is connected to its homing switch using more than one 34Mb/s connection. Then replacing two of these 34Mb/s connections by one 155Mb/s connection, as well as replacing a 34Mb/s interface card by a 155Mb/s interface card in the switch, again yields a feasible solution, for which the costs have not increased. Repeating this argument yields the first part of the result. The second part of our claim can be verified similarly. ■

Lemma 3.3.2 *(i). Consider the bin-packing problem for a demand node u as defined by constraints (3.9) and (3.10), and with objective function*

$$min \quad \sum_{j \in J_u} \lambda_j \cdot m_u^j \tag{3.22}$$

with $2 \cdot \lambda^{34} \geq \lambda^{155} > 0$. Let $(\bar{t}, \bar{m})$ represent an optimal solution for this bin-packing problem. If the conditions of Lemma 3.3.1 are satisfied, then there exists an optimal solution $(\bar{x}, \bar{y}, \bar{k}, \bar{l}, \bar{t}, \bar{m}, \bar{z}, \bar{n})$ for ATNIP with $\bar{t} = \bar{t}, \bar{m} = \bar{m}$.

(ii). Consider the bin-packing problem for a potential switch node $u \neq 0$ as defined by constraints (3.11) and (3.12), and with objective function

$$min \quad \sum_{j \in H_u} \lambda_j \cdot n_u^j \tag{3.23}$$

with $2 \cdot \lambda^{34} \geq \lambda^{155} > 0$. Let $(\tilde{z}, \tilde{n})$ represent an optimal solution for this bin-packing problem. If the conditions of Lemma 3.3.1 are satisfied, and there exists an optimal solution for ATNIP with a switch installed in node u, then there exists an optimal solution $(\bar{x}, \bar{y}, \bar{k}, \bar{l}, \bar{t}, \bar{m}, \bar{z}, \bar{n})$ for ATNIP with $\bar{z} = \tilde{z}, \bar{n} = \tilde{n}$.

Proof. Let $(\tilde{t}, \tilde{m})$ represent an optimal solution of the bin-packing problem, and let the optimal solution for ATNIP be denoted $(\bar{x}, \bar{y}, \bar{k}, \bar{l}, \bar{t}, \bar{m}, \bar{z}, \bar{n})$. Now suppose that $\tilde{m} \neq \bar{m}$. Define $\tilde{b}_u^\tau = \sum_{j \in J_u^\tau} \tilde{m}_u^j$ to be the number of connections of type τ used in the optimal bin-packing solution, and let $\bar{b}_u^\tau$ be defined similarly to be the number of connections of type τ used in the optimal ATNIP solution. We consider 8 cases.

(i). $\tilde{b}_u^{34} = \bar{b}_u^{34}, \tilde{b}_u^{155} > \bar{b}_u^{155}$; then a simple exchange argument makes it is easy to verify that $\tilde{m}$ cannot be an optimal solution for the bin-packing problem.

(ii). $\tilde{b}_u^{34} = \bar{b}_u^{34}, \tilde{b}_u^{155} < \bar{b}_u^{155}$; then $\bar{m}$ is not an optimal solution for ATNIP.

(iii). $\tilde{b}_u^{34} > \bar{b}_u^{34}, \tilde{b}_u^{155} = \bar{b}_u^{155}$; then $\tilde{m}$ is not an optimal solution for the bin packing problem.

(iv). $\tilde{b}_u^{34} < \bar{b}_u^{34}, \tilde{b}_u^{155} = \bar{b}_u^{155}$; then $\bar{m}$ is not an optimal solution for ATNIP.

(v). $\tilde{b}_u^{34} > \bar{b}_u^{34}, \tilde{b}_u^{155} > \bar{b}_u^{155}$; then $\tilde{m}$ is not an optimal solution for the bin packing problem.

(vi). $\tilde{b}_u^{34} < \bar{b}_u^{34}, \tilde{b}_u^{155} < \bar{b}_u^{155}$; then $\bar{m}$ is not an optimal solution for ATNIP.

(vii). $\tilde{b}_u^{34} > \bar{b}_u^{34}, \tilde{b}_u^{155} < \bar{b}_u^{155}$; from Lemma 3.3.1 it follows that we may assume w.l.o.g. that $\tilde{b}_u^{34} = 1, \bar{b}_u^{34} = 0$. Next, if $\bar{b}_u^{155} \geq \tilde{b}_u^{155} + 2$, then $\bar{b}_u$ cannot be optimal for ATNIP, so assume that $\bar{b}_u^{155} = \tilde{b}_u^{155} + 1$. But then it holds that replacing the connections $\bar{m}_u$ by connections $\tilde{m}_u$ yields a feasible solution for ATNIP with lower costs, hence $\bar{m}_u$ is not optimal for ATNIP.

(viii). $\tilde{b}_u^{34} < \bar{b}_u^{34}, \tilde{b}_u^{155} > \bar{b}_u^{155}$; from Lemma 3.3.1 it follows that we may assume w.l.o.g. that $\tilde{b}_u^{34} = 0, \bar{b}_u^{34} = 1$. Next, if $\bar{b}_u^{155} \leq \tilde{b}_u^{155} - 2$, then $\bar{b}_u$ cannot be optimal for the bin-packing problem, so assume that $\bar{b}_u^{155} = \tilde{b}_u^{155} - 1$. But then it holds that replacing the connections $\tilde{m}_u$ by connections $\bar{m}_u$ yields a feasible solution for the bin-packing problem with lower costs, hence $\tilde{m}_u$ is not optimal for the bin-packing problem.

This shows that indeed $\bar{m}_u = \tilde{m}_u$ must hold. Its also easy to see that $\bar{t} = \tilde{t}$, since the same assignment of the commodities can be used for both problems. Similarly, one can prove the second part of the Lemma. ■

Under the conditions of Lemma 3.3.1, the optimal values of the variables $t_u^{ij}, z_u^{ij}, m_u^j, n_u^j$ can thus be determined independently of the homing and routing decisions, by using the bin-packing problems as defined in Lemma 3.3.2. Hence, the optimal number of connections

between a demand node u and its homing switch (denoted b_u) can be obtained beforehand (although it requires solving a problem which is NP-hard in general, but relatively easy to solve for most real-life problem instances). Hence, $b_u^\tau = \sum_{j \in J_u^\tau} m_u^j$ with m_u^j the optimal value of the corresponding bin-packing problem. Similarly, the optimal number of connections between a switch and its homing switch (denoted p_u) can be determined beforehand: $p_u^\tau = \sum_{j \in H_u^\tau} n_u^j$ with n_u^j the optimal value of the corresponding bin-packing problem, and the sum over an empty set defined equal to zero. Thus, under the conditions of Lemma 3.3.2 ATNIP can be formulated as follows (denoted ATNIP* in the sequel):

$$\min \quad \sum_{w \in \mathcal{V}} K_w(k_w) + \sum_{e \in \mathcal{E}} L_e(\ell_e) \tag{3.24}$$

$$\text{s.t.} \quad \sum_{w \in X_u} x_{uw} = 1 \qquad \forall u \in \mathcal{V} \tag{3.25}$$

$$x_{u'w} \geq x_{uw} \qquad \forall u, u', w \in \mathcal{V} : w \in X_u, u' \in V(u,w) \backslash \{u\} \tag{3.26}$$

$$\sum_{w \in Y_u} y_{uw} = x_{uu'} \qquad \forall u \in \mathcal{V} \setminus \{0\} \tag{3.27}$$

$$y_{uw} \leq x_{u'w} \qquad \forall u, u', w \in \mathcal{V} : w \in Y_u, u' \in V(u,w) \backslash \{u\} \tag{3.28}$$

$$k_w^\tau = \sum_{u \in \mathcal{V}} (b_u^\tau x_{uw} + p_u^\tau y_{uw}) + p_w^\tau x_{ww} \qquad \forall w \in \mathcal{V}, \tau = 1, \ldots, t \tag{3.29}$$

$$\ell_e^\tau = \sum_{u,w \in \mathcal{V} : e \in E(u,w)} (b_u^\tau x_{uw} + p_u^\tau y_{uw}) \qquad \forall e \in \mathcal{E}, \tau = 1, \ldots, t \tag{3.30}$$

$$x_{uw} \in \{0,1\} \qquad \forall u \in \mathcal{V}, \forall w \in X_u \tag{3.31}$$

$$y_{uw} \in \{0,1\} \qquad \forall u \in \mathcal{V} \setminus \{0\}, \forall w \in Y_u \tag{3.32}$$

$$k_w^\tau,\ \ell_e^\tau \geq 0 \qquad \forall w \in \mathcal{V},\ \forall e \in \mathcal{E},\ \tau = 1, \ldots, t \tag{3.33}$$

Unless stated otherwise, in the sequel we will assume that the conditions of Lemma 3.3.2 are satisfied, hence the latter formulation of ATNIP* will be used.

Now we show that the decision version of ATNIP* is NP-complete, which implies that an exact algorithm running in pseudo-polynomial time is the best one may expect, unless P=NP. To do so, we state the following problem definitions:

SUBSET SUM (see Garey and Johnson [37])
INSTANCE: Finite set A, size $s(a) \in \mathbb{N}$ for every $a \in A$, and a positive integer D.
QUESTION: Is there a subset $A' \subseteq A$ such that the sum of the sizes of the elements in A' is equal to D?

ATNIP*
INSTANCE: Tree $G = (\mathcal{V}, \mathcal{E})$, an integer B, a number $t \in \mathbb{N}$, a demand $b_u^\tau \in \mathbb{N}$ for every $u \in \mathcal{V}, \tau = 1, \ldots, t$, a number $p_u \in \mathbb{N}$ for every $u \in \mathcal{V}, \tau = 1, \ldots, t$, and cost functions $K_w(k_w) : \Pi_\tau \{0, \ldots, \gamma^\tau \cdot B\} \to \mathbb{N}$, for all $w \in \mathcal{V}$ and $L_e(\ell_e) : \Pi_\tau \{0, \ldots, \gamma^\tau \cdot B\} \to \mathbb{N}$, for all $e \in \mathcal{E}$, and a positive integer F.
QUESTION: Does there exist a solution (x, k, ℓ) for ATNIP as defined by (3.24)–(3.33) with objective value at most F?

Theorem 3.3.1 *ATNIP* is NP-complete (under the assumption that for each k_w, ℓ_e and F, $\sum_w K_w(k_w) + \sum_e L_e(\ell_e) \leq F$ can be verified in polynomial time).*

Proof. It can easily be checked that ATNIP* is in NP (given the mild assumption that the objective value can be verified in polynomial time). Hence it suffices to show that SUBSET SUM reduces to ATNIP*. Given an instance for SUBSET SUM, we define an instance for ATNIP* as follows. Let $\mathcal{V} = A \cup \{0\}$ contain a node for each item in A and a node which will be the root of the tree. Let $\mathcal{E} = \{\{0, a\} | a \in A\}$ be the set of edges where each node in A is connected to the root. Next, we let $t = 1$ and $\gamma^t = 1$ (only one type of connection and the number of slots equals the number of connections which can be connected to a switch). Define $b_a = s(a)$ for all $a \in A$ and $b_0 = 0$, $p_a = 0$ for all $a \in A \cup \{0\}$, $B = \sum_{a \in A} s(a)$, and $F = 1$. Finally, edge costs $L_e(\ell_e) = 0$ for all $e \in \mathcal{E}$, and $K_w(k_w) = 0$ for $w \neq 0$, $K_w(k_w) = 1$ for $w = 0, k_w = D$, and $K_w(k_w) = 2$ for $w = 0, k_w \neq D$. Note that this transformation can be done in polynomial time. It remains to show that the two problem instances are equivalent.

First, assume that there exists a subset $A' \subseteq A$ with an aggregated size equal to D. By installing a large switch in the nodes corresponding to items in $A \setminus A'$, as well as in the root, we obtain a feasible solution for ATNIP* with objective value equal to one. Conversely, if there exists a solution for ATNIP* with solution value less than or equal to one, it must be equal to one, since the switch costs in the root are at least one. But then the load on the switch in the root must equal D, which in turn implies that the set of nodes in which no switch is installed has a cumulative demand equal to D. ■

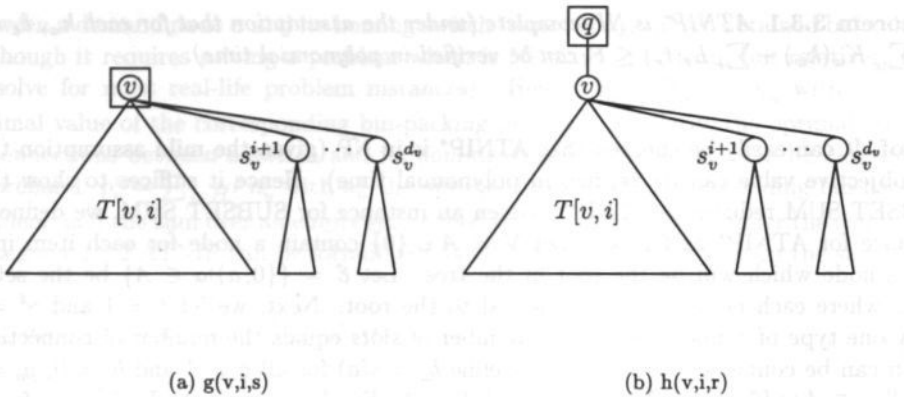
3.4 Defining Subproblems for ATNIP

In this section we introduce two families of subproblems which are defined on subtrees $T[v, i]$. These subtrees were introduced by Johnson and Niemi [51] to improve the running time of a dynamic programming algorithm for partially ordered knapsack problems. The main idea is that a subtree $T[v, i]$ can be decomposed into two smaller subtrees $T[v, i-1]$ and $T[s^i_v, d_{s^i_v}]$ (provided $i \geq 1$). By combining optimal solutions for problems defined on the latter subtrees, one may be able to obtain an optimal solution for the former. The main difficulty (as is the case for all dynamic programming algorithms) is to find an appropriate parameterization of the problem at hand.

The first family of subproblems we define are denoted by $g(v, i, s)$. They represent the minimal costs restricted to the subtree $T[v, i]$, among all solutions in which a switch is installed in node v. Furthermore, the vector $s \in \mathbb{N}^t$ denotes the connections between the switch in node v and nodes in $V[v, d_v] \setminus V[v, i]$ (see Figure 3.2(a)). More formally, for (v, i) with $v \in \mathcal{V}$, $0 \leq i \leq d_v$, $s \geq 0$ and $s + b_v + p_v \ll B$ we define $g(v, i, s)$ to be

$$\min \quad \sum_{w \in V[v,i] \setminus \{v\}} K_w(k_w) + K_v(k_v + s) + \sum_{e \in E[v,i]} L_e(\ell_e) \tag{3.34}$$

$$\text{s.t.} \quad \sum_{w \in X_u \cap V[v,i]} x_{uw} = 1 \qquad \forall u \in V[v, i] \tag{3.35}$$



(a) g(v,i,s) (b) h(v,i,r)

Figure 3.2: Graphical Representation of Subproblems

$$x_{u'w} \geq x_{uw} \qquad \forall u \in V[v,i], \forall w \in X_u \cap V[v,i], \forall u' \in V(u,w) \setminus \{u\} \tag{3.36}$$

$$\sum_{w \in Y_u \cap V[v,i]} y_{uw} = x_{uu} \qquad \forall u \in V[v,i] \setminus \{v\} \tag{3.37}$$

$$y_{uw} \leq x_{u'w} \qquad \forall u \in V[v,i], \forall w \in Y_u \cap V[v,i], \forall u' \in V(u,w) \setminus \{u\} \tag{3.38}$$

$$k_w^\tau = \sum_{u \in V[v,i]} (b_u^\tau x_{uw} + p_u^\tau y_{uw}) + p_w^\tau x_{ww} \qquad \forall w \in V[v,i], \tau = 1, \ldots, t \tag{3.39}$$

$$\ell_e^\tau = \sum_{u,w \in V[v,i]: e \in E(u,w)} (b_u^\tau x_{uw} + p_u^\tau y_{uw}) \qquad \forall e \in E[v,i], \tau = 1, \ldots, t \tag{3.40}$$

$$x_{uw} \in \{0,1\} \qquad \forall u \in V[v,i], \forall w \in V[v,i] \cap X_u \tag{3.41}$$

$$y_{uw} \in \{0,1\} \qquad \forall u \in V[v,i] \setminus \{v\}, \forall w \in V[v,i] \cap Y_u \tag{3.42}$$

$$k_w^\tau,\ \ell_e^\tau \geq 0 \qquad \forall w \in V[v,i], \forall e \in E[v,i], \tau = 1, \ldots, t \tag{3.43}$$

The second family of subproblems $h(v,i,r)$ represents the minimal costs restricted to the subtree $T[v,i]$, among all solutions for which no switch is installed in node v. The vector r represents the connections between nodes in $T[v,i]$ and a switch located outside $T[v,i]$. In order to have such a homing node for node v (as well as the endpoint outside $T[v,i]$ for the connections of the vector r), we introduce an artificial node q to be a predecessor of node v, in which a switch is installed (see Figure 3.2(b)). More formally, for (v,i) with $v \in \mathcal{V}$, $0 \leq i \leq d_v$ and $b_v \leq r \ll B$ we define $h(v,i,r)$ to be

$$\min \quad \sum_{w \in V[v,i]} K_w(k_w) + \sum_{e \in E[v,i]} L_e(\ell_e) \tag{3.44}$$

$$\text{s.t.} \quad \sum_{w \in X_u \cap V[v,i]} x_{uw} + x_{uq} = 1 \qquad \forall u \in V[v,i] \cup \{q\} \tag{3.45}$$

$$x_{vv} = 0 \tag{3.46}$$

$$x_{u'w} \geq x_{uw} \qquad \forall u \in V[v,i], \forall w \in (X_u \cap V[v,i]) \cup \{q\}, \forall u' \in V(u,w) \setminus \{u\} \tag{3.47}$$

$$\sum_{w \in Y_u \cap V[v,i]} y_{uw} + y_{uq} = x_{uu} \qquad \forall u \in V[v,i] \tag{3.48}$$

$$y_{u'w} \leq x_{uw} \qquad \forall u \in V[v,i], \forall w \in (Y_u \cap V[v,i]) \cup \{q\}, \forall u' \in V(u,w) \setminus \{u\} \tag{3.49}$$

$$k_w^\tau = \sum_{u \in V[v,i]} (b_u^\tau x_{uw} + p_u^\tau y_{uw}) + p_w^\tau x_{ww} \qquad \forall w \in V[v,i] \cup \{q\}, \tau = 1, \ldots, t \tag{3.50}$$

$$\ell_e^\tau = \sum_{u,w \in V[v,i] \cup \{q\} : e \in E(u,w)} (b_u^\tau x_{uw} + p_u^\tau y_{uw}) \qquad \forall e \in E[v,i] \cup \{v\}, \tau = 1, \ldots, t \tag{3.51}$$

$$\ell_v^\tau = r^\tau \qquad \tau = 1, \ldots, t \tag{3.52}$$

$$x_{uw} \in \{0,1\} \qquad \forall u \in V[v,i], \forall w \in (V[v,i] \cap X_u) \cup \{q\} \tag{3.53}$$

$$y_{uw} \in \{0,1\} \qquad \forall u \in V[v,i], \forall w \in (V[v,i] \cap Y_u) \cup \{q\} \tag{3.54}$$

$$k_w^\tau,\ \ell_e^\tau \geq 0 \qquad \forall w \in V[v,i], \forall e \in E[v,i], \tau = 1, \ldots, t \tag{3.55}$$

During our dynamic programming algorithm, solutions for the subtrees $T[v, i-1]$ and $T[s_v^i, d_{s_v^i}]$ will be combined to obtain a solution for subtree $T[v,i]$. We will therefore focus on the relations between the subproblems we have just defined. In Section 3.4.1 we show how the problem $g(v,i,s)$ for $i > 0$ can be decomposed to problems on subtrees $T[v, i-1]$ and $T[s_v^i, d_{s_v^i}]$, whereas Section 3.4.2 reports on similar relations for $h(v,i,r)$. In Section 3.4.3 it is shown how $g(v,i,s)$ and $h(v,i,r)$ should be determined in case $i = 0$, i.e. if the subtree consists of a single node. These relations form the starting point of the dynamic programming algorithm for ATNIP*.

3.4.1 Recursive Relations for $g(v,i,s)$.

The set of feasible solutions for $g(v,i,s)$ can be partitioned into a set of solutions for which $x_{s_v^i s_v^i} = 1$ (representing the situation in which a switch is installed in node s_v^i), and a set of solutions for which $x_{s_v^i s_v^i} = 0$ (no switch is installed in node s_v^i). Let $\alpha \in \mathbb{N}^t$ denote the number of connections between a switch in node v and nodes in $V[v,i] \setminus V[v, i-1]$ in a feasible solution for ATNIP with $x_{vv} = 1$, hence for $g(v,i,s)$. For a feasible solution in the case that $x_{s_v^i s_v^i} = 1$, α must equal $p_{s_v^i}$. The set of feasible solutions in the case that $x_{s_v^i s_v^i} = 0$, can be partitioned into smaller sets by considering all possible values of α. Because of the partition, the optimal value for $g(v,i,s)$ can be determined as the minimum of the optimal values over all subsets of feasible solutions. Therefore, one can show that the optimal value for $g(v,i,s)$ can be obtained from the aforementioned subsets of feasible solutions.

Lemma 3.4.1 *Consider (v,i) with $v \in \mathcal{V}$ and $i > 0$. Let (x,k,ℓ) be an optimal solution*

for $g(v,i,s)$ with $x_{s_v^i s_v^i} = 1$. Then

$$g(v,i,s) = g(s_v^i, d_{s_v^i}, 0) + g(v, i-1, s + p_{s_v^i}) + L_{s_v^i}(p_{s_v^i}) \tag{3.56}$$

Proof. From $x_{s_v^i s_v^i} = 1$ it follows that $x_{uv} = 0$ for all $u \in V[s_v^i, d_{s_v^i}]$, and $y_{uv} = 0$ for all $u \in V[s_v^i, d_{s_v^i}] \setminus \{s_v^i\}$. But then constraints (3.35)–(3.43) of $g(v,i,s)$ can be formulated as follows:

$$\sum_{w \in X_u \cap V[v,i-1]} x_{uw} = 1 \qquad \forall u \in V[v, i-1]$$

$$\sum_{w \in X_u \cap V[s_v^i, d_{s_v^i}]} x_{uw} = 1 \qquad \forall u \in V[s_v^i, d_{s_v^i}]$$

$$x_{u'w} \geq x_{uw} \qquad \forall u \in V[v,i-1], \forall w \in X_u \cap V[v,i-1]$$
$$\forall u' \in V(u,w) \setminus \{u\}$$

$$x_{u'w} \geq x_{uw} \qquad \forall u \in V[s_v^i, d_{s_v^i}], \forall w \in X_u \cap V[s_v^i, d_{s_v^i}]$$
$$\forall u' \in V(u,w) \setminus \{u\}$$

$$\sum_{w \in Y_u \cap V[v,i-1]} y_{uw} = x_{uu} \qquad \forall u \in V[v,i-1] \setminus \{v\}$$

$$\sum_{w \in Y_u \cap V[s_v^i, d_{s_v^i}]} y_{uw} = x_{uu} \qquad \forall u \in V[s_v^i, d_{s_v^i}] \setminus \{s_v^i\}$$

$$y_{s_v^i v} = x_{s_v^i s_v^i} = 1$$

$$y_{uw} \leq x_{u'w} \qquad \forall u \in V[v,i-1], \forall w \in Y_u \cap V[v,i-1]$$
$$\forall u' \in V(u,w) \setminus \{u\}$$

$$y_{uw} \leq x_{u'w} \qquad \forall u \in V[s_v^i, d_{s_v^i}], \forall w \in Y_u \cap V[s_v^i, d_{s_v^i}]$$
$$\forall u' \in V(u,w) \setminus \{u\}$$

$$k_w^\tau = \sum_{u \in V[v,i-1]} (b_u^\tau x_{uw} + p_u^\tau y_{uw}) + p_w^\tau x_{ww} \qquad \forall w \in V[v,i-1] \setminus \{v\}, \tau = 1, \ldots, t$$

$$k_w^\tau = \sum_{u \in V[s_v^i, d_{s_v^i}]} (b_u^\tau x_{uw} + p_u^\tau y_{uw}) + p_w^\tau x_{ww} \qquad \forall w \in V[s_v^i, d_{s_v^i}], \tau = 1, \ldots, t$$

$$k_v^\tau = \sum_{u \in V[v,i-1]} (b_u^\tau x_{uw} + p_u^\tau y_{uw}) + p_v^\tau x_{vv} + p_{s_v^i}^\tau \qquad \tau = 1, \ldots, t$$

$$\ell_e^\tau = \sum_{u,w \in V[v,i-1]: e \in E(u,w)} (b_u^\tau x_{uw} + p_u^\tau y_{uw}) \qquad \forall e \in E[v,i-1], \tau = 1, \ldots, t$$

$$\ell_e^\tau = \sum_{u,w \in V[s_v^i, d_{s_v^i}]: e \in E(u,w)} (b_u^\tau x_{uw} + p_u^\tau y_{uw}) \qquad \forall e \in E[s_v^i, d_{s_v^i}], \tau = 1, \ldots, t$$

$$\ell_{s_v^i}^\tau = p_{s_v^i}^\tau \qquad \tau = 1, \ldots, t$$

$$x_{uw} \in \{0,1\} \qquad \forall u \in V[v,i-1]$$
$$\forall w \in V[v,i-1] \cap X_u$$

$$x_{uw} \in \{0,1\} \qquad \forall u \in V[s_v^i, d_{s_v^i}]$$
$$\forall w \in V[s_v^i, d_{s_v^i}] \cap X_u$$

$$y_{uw} \in \{0,1\} \qquad \forall u \in V[v,i-1] \setminus \{v\}$$
$$\forall w \in V[v,i-1] \cap Y_u$$

$$y_{uw} \in \{0,1\} \qquad \forall u \in V[s_v^i, d_{s_v^i}] \setminus \{s_v^i\}$$
$$\forall w \in V[s_v^i, d_{s_v^i}] \cap Y_u$$
$$k_w^\tau,\ \ell_e^\tau \geq 0 \qquad \forall w \in V[v, i-1]$$
$$\forall e \in E[v, i-1]\tau = 1, \ldots, t$$
$$k_w^\tau,\ \ell_e^\tau \geq 0 \qquad \forall w \in V[s_v^i, d_{s_v^i}]$$
$$\forall e \in E[s_v^i, d_{s_v^i}]\tau = 1, \ldots, t$$

This shows that the constraints can be separated into two sets, one set containing the constraints on the subtree $T[v, i-1]$, and one set containing the constraints on the subtree $T[s_v^i, d_{s_v^i}]$. Since the objective function is also separable, it directly follows that the problem $g(v,i,s)$ decomposes into the problem $g(s_v^i, d_{s_v^i}, 0)$ on subtree $T[s_v^i, d_{s_v^i}]$, problem $g(v, i-1, s+p_{s_v^i})$ on subtree $T[v, i-1]$, and the individual edge s_v^i with $\ell_{s_v^i} = p_{s_v^i}$. ■

Lemma 3.4.2 *Consider (v,i) with $v \in \mathcal{V}$ and $i > 0$. Let (x,k,ℓ) be an optimal solution for $g(v,i,s)$ with $x_{s_v^i s_v^i} = 0$ and $\ell_{s_v^i} = \alpha$. Then*

$$g(v,i,s) \;=\; h(s_v^i, d_{s_v^i}, \alpha) + g(v, i-1, s+\alpha) + L_{s_v^i}(\alpha) \tag{3.57}$$

Proof. Similar. ■

3.4.2 Recursive Relations for $h(v,i,r)$.

In this section we use a similar partition of the set of feasible solutions for $h(v,i,r)$ to obtain its optimal value and solution. Since the proofs of the lemmas are similar to the one in the previous section, they are left to the reader.

Lemma 3.4.3 *Consider (v,i) with $v \in \mathcal{V}$ and $i > 0$. Let (x,k,ℓ) be an optimal solution for $h(v,i,r)$ with $x_{s_v^i s_v^i} = 1$. Then*

$$h(v,i,r) \;=\; g(s_v^i, d_{s_v^i}, 0) + h(v, i-1, r-p_{s_v^i}) + L_{s_v^i}(p_{s_v^i}) \tag{3.58}$$

Lemma 3.4.4 *Consider (v,i) with $v \in \mathcal{V}$ and $i > 0$. Let (x,k,ℓ) be an optimal solution for $h(v,i,r)$ with $x_{s_v^i s_v^i} = 0$ and $\ell_{s_v^i} = \alpha$. Then*

$$h(v,i,r) \;=\; h(s_v^i, d_{s_v^i}, \alpha) + h(v, i-1, r-\alpha) + L_{s_v^i}(\alpha) \tag{3.59}$$

3.4.3 Starting Point of Dynamic Programming Algorithm.

In this section we show how the coefficients $g(v,i,s)$ and $h(v,i,r)$ can be determined if $i = 0$, i.e. if the tree $T[v,i]$ consists of a single node. Since the parameterizations are defined in such a way that it is completely clear what the incoming or outgoing load into the tree is, these coefficients can be determined very easily. Together with the relations as defined in the above, these relations form the building blocks of our dynamic programming algorithm.

Proposition 3.4.1 *Consider (v,i) with $v \in \mathcal{V}$ and $i = 0$. If $s \in \mathbb{N}^t$ is such that the inequality $s + b_v + p_v \ll B$ holds, then*

$$g(v,i,s) = K_v(s + b_v + p_v) \tag{3.60}$$

Proof. It is easy to see that under the condition posed in the proposition there exists a solution for $g(v,0,s)$. Since $V[v,0] = \{v\}$, it follows that $x_{vv} = 1$. Furthermore,

$$\begin{aligned} k_v^\tau &= \textstyle\sum_{u \in V[v,0]} (b_u^\tau \cdot x_{uv} + p_u^\tau \cdot y_{uv}) + p_v^\tau \cdot x_{vv} \\ &= b_v^\tau \cdot x_{vv} + p_v^\tau \cdot x_{vv} \\ &= b_v^\tau + p_v^\tau \end{aligned}$$

The objective function in (3.34) thus amounts to $K_v(s + b_v + p_v)$. ∎

Proposition 3.4.2 *Consider (v,i) with $v \in \mathcal{V}$ and $i = 0$. Then*

$$h(v,i,r) = \begin{cases} K_v(0) & \text{if } r = b_v \\ \infty & \text{otherwise} \end{cases} \tag{3.61}$$

Proof. From $V[v,0] = \{v\}$, $x_{vv} = 0$ and (3.52) it follows that $r = b_v$. Finally, the objective function in (3.44) amounts to $K_v(0)$. ∎

3.5 An $\mathcal{O}(nB^4)$ Dynamic Programming Algorithm for ATNIP

Based on the relations that are derived in the preceding section, we are now able to formulate a dynamic programming algorithm. Recall that $g(v,i,s)$ is only defined for (v,i) with $v \in \mathcal{V}$, $0 \leq i \leq d_v$ and $0 \leq s$ and $s + b_v + p_v \ll B$. Similarly, $h(v,i,r)$ is only defined on (v,i) pairs with $v \neq 0$ and $b_v \leq r \ll B$. The dynamic programming algorithm

as stated here does not keep track of the optimal solution itself, but standard ideas from dynamic programming can be used to do so.

DYNAMIC PROGRAMMING ALGORITHM FOR ATNIP

forall (v,i,s) with $v \in \mathcal{V}$, $0 \le i \le d_v$ and $0 \le s$ and $s + b_v + p_v \ll B$ **do**
  $g(v,i,s) = \infty$; /* *initialization* g */
forall (v,i,r) with $v \in \mathcal{V}$, $0 \le i \le d_v$ and $b_v \le r \ll B$ **do**
  $h(v,i,r) = \infty$; /* *initialization* h */
forall $v = n$ **downto** 0 **do begin**
  forall s with $0 \le s$ and $s + b_v + p_v \ll B$ **do**
    $g(v,0,s) = K_v(s + b_v + p_v)$;
  if $(v \neq 0)$ **then**
    $h(v,0,r_v) = K_v(0)$;
  forall $i = 1$ **to** d_v **do begin**
    forall s with $0 \le s$ and $s + b_v + p_v \ll B$ **do**
      $g(v,i,s) = \min\{g(s_v^i, d_{s_v^i}, 0) + g(v,i-1,s+p_{s_v^i}) + L_{s_v^i}(p_{s_v^i}),$
        $\min_\alpha\{h(s_v^i, d_{s_v^i}, \alpha) + g(v,i-1,s+\alpha) + L_{s_v^i}(\alpha)\}\}$;
    forall r with $b_v \le r \ll B$ **do**
      $h(v,i,r) = \min\{g(s_v^i, d_{s_v^i}, 0) + h(v,i-1,r-p_{s_v^i}) + L_{s_v^i}(p_{s_v^i}),$
        $\min_\alpha\{h(s_v^i, d_{s_v^i}, \alpha) + h(v,i-1,r-\alpha) + L_{s_v^i}(\alpha)\}\}$;
  end;
end;

When determining the coefficients $g(v,i,s)$ and $h(v,i,r)$ for $i > 0$, one needs to know all possible values of α which may occur. Weak bounds can be given, such as $\alpha \ge 0$, since obviously $\ell_{s_v^i}$ must be nonnegative, and $\alpha \ll B$, since all connections which pass the same edge are connected to the same switch, and hence, must fit the capacity of that switch. Of course, the running time of the algorithm can be improved by exploiting better bounds. Since α must at least encompass the connections from demand node s_v^i, $\alpha \ge b_{s_v^i}$ would be a valid lower bound. Furthermore, when determining $g(v,i,s)$, all connections on edge s_v^i must fit on the switch in node v. But the same holds for connections from vector s, the demand from node v itself, and the connections between the switch in v and its homing switch. Hence, $\alpha + s + b_v + p_v \ll B$ yields a valid upper bound on α for this case. Finally, when determining $h(v,i,r)$, the connections on edge s_v^i together with the demand from node v are only a part of the total number of connections r, hence, $\alpha + b_v \le r$ imply a valid upper bound for α in this case.

In practical applications the number of different types of connections is usually small ($t = 2$). Furthermore if the applied cost structure is as indicated in Section 3.2, the

general cost figures $K_w(k_w)$ and $L_e(\ell_e)$ can be determined easily (that is in small running time). The overall running time of the algorithm then equals $\mathcal{O}(nB^4)$, since the number of $T[v,i]$ trees we consider is $\mathcal{O}(n)$, for each tree we determine $\mathcal{O}(B^2)$ g-coefficients and $\mathcal{O}(B^2)$ h-coefficients. Moreover, each of these coefficients can be determined in $\mathcal{O}(B^2)$ time. The required storage space follows directly from the number of coefficients to be determined and is $\mathcal{O}(nB^2)$. This is formally stated in the following theorem, which is the main result of this chapter.

Theorem 3.5.1 *Consider ATNIP* as defined by (3.4)–(3.18), and with objective function defined by (3.20) and (3.21). If $t = 2$ (two types of connections and interface cards) and $\theta \leq |\mathcal{V}|$ (the number of switches bounded by the number of nodes in the tree), then ATNIP* can be solved by a pseudo-polynomial time dynamic programming algorithm that requires $\mathcal{O}(nB^4)$ time, and $\mathcal{O}(nB^2)$ storage space.*

When the algorithm is used as a tool in the planning phase of ATM network layout, there are several features which may be added to the algorithm. First, if one is interested in the optimal solution in which certain switches are already (pre)installed, one can easily incorporate this into the algorithm by adopting the cost function $K_w(k_w)$ for these specific nodes. A similar technique can of course be applied to prohibit the installation of switches in a node. Overall, this gives the possibility to incorporate partial solutions as part of the input. This can very well help to resolve many sensitivity questions which are extremely important in the planning of network structures.

Secondly, given a solution, it is easy to show the actual number of slots being used in a switch, as well as the number of Mb/s flowing over a set of connections. Hence, given a network structure, one can determine to what extend switches and connections are being used. This utilization level of the different components of the network may give insight into the stability of the network structure for future demand.

Thirdly, when one determines the optimal solution for the tree as a whole, but for different amounts of traffic from outside the tree (i.e. not only zero), this amount of traffic from outside the tree may be viewed as the number of connections incident to the root node which are used for routing in the Backbone. If the optimal solution is insensitive for this amount of traffic, this gives a justification for the approach to solve ATM network problems for the Backbone and trees separately.

Finally, in practice the number of connections used is, although small (usually equal to two), of great importance for the running time of the algorithm, as it is proportional to the number of types available. By only allowing one type of connection (say the largest capacity), an approximation algorithm can be made which from a theoretical point of view may have a bad performance, but from a practical point of view can be very useful. By only considering one type of connection the running time decreases significantly, whereas the solutions obtained are feasible and usually capable of processing a somewhat larger demand than the demand which forms the input of the algorithm.

3.6 Computational Results

To test our algorithm on real-life instances from the Dutch telecommunication company, we implemented our algorithm in C++ on a DEC 2100 A500MP workstation with 128Mb of internal memory. All problem instances we consider have a cost structure as the one proposed in Section 3.2 and Section 3.3. Table 3.1 reports on eight problem instances, for which three different switches were available, with capacities 3,10 and 20 slots, respectively. Table 3.2 reports on the same instances but with the largest switch capacity being

Table 3.1: Computational results for the instances from KPN (time measured in sec.)

problem	n	dp_aso_1 (s)	value
aso_1	12	0.37	92
aso_2	26	1.14	141
aso_3	29	1.71	161
aso_4	35	1.45	178
aso_5	28	0.96	151
aso_6	12	0.67	85
aso_7	22	1.51	147
aso_8	31	1.15	146

Table 3.2: Computational results for the instances from KPN with larger capacity switch.

problem	n	dp_aso_1 (s)	value
aso_1	12	11.59	92
aso_2	26	29.57	141
aso_3	29	50.59	161
aso_4	35	37.83	178
aso_5	28	25.14	151
aso_6	12	18.53	85
aso_7	22	42.51	147
aso_8	31	31.68	146

50 instead of 20 (B increases). The results show that B is indeed an important determinant for the running time of the algorithm. The optimal solutions have not changed, which only indicates that capacity on the switches was not a restraining factor.

From the above results it follows that if extremely large trees are considered, together with even larger capacity switches, the running times of the algorithm might become

unacceptable. Therefore we have also implemented an algorithm *dp_aso_2* in which only one type of connection (155Mb/s connections) is considered (as explained in Section 3.5). This relaxation causes the number of g and h-coefficients which have to be determined to be reduced significantly. Table 3.3 shows that the running times of the algorithm behave accordingly. More importantly, although the algorithm does no longer need to provide an optimal solution, the solutions found by the algorithm proved to be very similar to the optimal solutions and still useful in practice. Finally, some larger trees were considered. Table 3.4 states the results for these larger instances, where again three switch capacities were considered, with capacities 3, 10, 30, respectively.

Table 3.3: Computational results for the instances from KPN with only one type of connection.

problem	n	dp_aso_2 (s)
aso_1	12	0.02
aso_2	26	0.03
aso_3	29	0.03
aso_4	35	0.02
aso_5	28	0.02
aso_6	12	0.02
aso_7	22	0.02
aso_8	31	0.03

Table 3.4: Computational results for the instances from KPN with larger capacity switch.

problem	n	dp_aso_1 (s)
aso_17	34	9.05
aso_24	61	12.90
aso_58	59	10.74
aso_68	43	8.79

In conclusion, the algorithmic ideas presented in this chapter prove to be very useful in practice. The algorithm yields optimal solutions for real-life problem instances, which are of significant problem size. Moreover, additional features can be added to the planning tool if desired, making the tool even more attractive as a decision support system in the network design planning phase.

Chapter 4

Lifting Valid Inequalities for the Precedence Constrained Knapsack Problem

4.1 Introduction

Many situations arise in practice where certain decisions can only be implemented or performed if specific other measures are taken. In order to analyze such situations by means of a mathematical model, one needs ways to model these implications. Precedence constraints are a natural way to do this, and in fact both the mathematical models in Chapter 2 and Chapter 3 contain precedence constraints to model the restriction that if a node in the tree network "homes on" a concentrator in another node, then all nodes on the path between the two nodes must home on that concentrator as well (see Chapter 2 for this terminology).

Apart from precedence constraints many of such telecommunication network design models also contain knapsack constraints to represent the limited capacity of concentrators (or any arbitrary telecommunication device) located in the network. For instance, Aghezzaf, Magnanti and Wolsey [3] consider a problem on a tree network where each node of the tree has a demand for telecommunication service which can be satisfied by a telecommunication device located in the root of the tree. The goal of the problem is to maximize the number of nodes whose demand can be satisfied given the limited capacity of the device (knapsack constraint) and under the restriction that a node's demand can only be satisfied if the demands of all nodes on the path from that specific node to the telecommunication device are satisfied as well (precedence constraint). Furthermore Shaw [69] and Cho and Shaw [23] employ a decomposition technique to tackle certain network design problems, for which subproblem that remains to be solved consist of a knapsack constraint and precedence constraints (see Cho and Shaw [22] and Shaw and Cho [70]).

In this chapter, based on Van de Leensel, van Hoesel and van de Klundert [76], we therefore study the precedence constrained knapsack problem (PCKP) which contains

both a knapsack constraint and a number of precedence constraints. The remainder of this introductory section discusses PCKP, the literature on its polyhedral structure, and the contributions of the research presented in this chapter.

Consider the standard 0–1 knapsack problem. In this problem, there is a set V of items, $V = \{1, 2, ..., n\}$ and a knapsack with capacity $b \in \mathbb{Z}^+$. Each item $i \in V$ has a value $c_i \in \mathbb{Z}$, and a weight $w_i \in \mathbb{Z}^+$. The problem is to find a maximum value subset of the set of items whose total weight does not exceed the knapsack capacity. This is modeled in the following integer linear programming formulation:

$$\max \quad \sum_{i=1}^{n} c_i x_i \tag{4.1}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} a_i x_i \leq b \tag{4.2}$$

$$x_i \in \{0, 1\} \qquad i = 1, ..., n \tag{4.3}$$

The knapsack problem has received considerable attention, not only because it has several important applications in itself, but also because it arises as a substructure in many combinatorial optimization problems.

This chapter studies the precedence constrained knapsack problem, which generalizes the knapsack problem by including a partial order on the items. We say that there is a precedence constraint *from* item i *to* item j if item j can be included in the knapsack only if item i is included. Thus, $x_i = 0 \Rightarrow x_j = 0$. The set of precedence constraints can be represented by a directed graph $D(V, A)$, where the node set V is the set of items, and each precedence constraint is represented by a directed arc in A. The precedence constraints are thus given by

$$x_i \geq x_j \qquad (i, j) \in A \tag{4.4}$$

The precedence constrained knapsack problem (PCKP) is now formulated by (4.1)-(4.4). In this chapter, we are interested in facet defining inequalities for the precedence constrained knapsack polytope, and more specifically, the complexity of obtaining these facets using lifting techniques.

As is the case for the standard 0–1 knapsack problem, PCKP is an interesting problem in itself, which arises naturally as a substructure in several other combinatorial problems (apart from the ones in telecommunication network design). Consider for instance a tool management problem that arises in automated manufacturing, where each part requires a specific set of tools in order to be processed. Hence, a part can only be processed on a machine if the required tools are loaded. In our model this would correspond to a precedence constraint from tool i to job j if tool i is required to process job j. The knapsack constraint stems from the limited capacity of the tool magazine. Crama [26] and Stecke and Kim [72] discuss several problems containing both knapsack and precedence constraints in the context of tool management, and provide pointers to literature on related combinatorial problems, of which we mention only a few here.

Mamer and Shogan [57] and Hwan and Shogan [47] consider capital constrained repair kit selection problems, which also have both knapsack and precedence constraints. A similar formulation arises in strip mining applications (see Johnson and Niemi [51]). Given a geological report on the expected contents of certain minerals in different layers of the earth, the mining company would like to develop a digging plan in order to maximize its profits. If digging in lower layers of the earth seems profitable due to a high concentration of the minerals, then this is only possible if digging in higher layers is performed as well (precedence constraints).

Garey and Johnson [37] prove that the decision version of PCKP is NP-complete in the strong sense, but solvable in pseudo-polynomial time if the underlying precedence graph is a tree (see Johnson and Niemi [51] for a dynamic programming algorithm). Hence, in order to solve the general PCKP to optimality, a further understanding of the structure of the precedence constrained knapsack polytope can be expected to accelerate general integer programming schemes, as it has led to more powerful exact solution methods for standard 0–1 knapsack problems (see Crowder, Johnson and Padberg [27]). For polyhedral results on the standard 0–1 knapsack problem we refer to Balas [10], Balas and Zemel [11] and Zemel [80]. Hartvigsen and Zemel [46] discuss the complexity of the recognition of (lifted) valid knapsack inequalities.

As is observed by Boyd [19], problems which are defined entirely by precedence constraints can be solved using standard LP-techniques, since a set of precedence constraints itself defines a totally unimodular matrix and hence, a polyhedron with integral vertices. For the PCKP, Boyd [19] analyzes two classes of valid inequalities arising from K-covers and (1,k)-configurations. He identifies conditions under which these inequalities define facets of a lower dimensional polytope, in which case lifting may lead to a facet of the precedence constrained knapsack polytope itself. Park and Park [64] consider a special case of K-covers which they refer to as minimal induced covers. In general, inequalities arising from minimal induced covers will not define facets of the precedence constrained knapsack polytope. Park and Park [64] consider a lifting technique to obtain valid inequalities. The reader may observe that all of the aforementioned classes of valid inequalities are natural extensions of classes of inequalities for the ordinary knapsack problem.

In this chapter, we present various new results on facets of the PCKP-polytope. Section 4.2 introduces some notation and assumptions used throughout the chapter. In Section 4.3 we state a class of lifting orders, which guarantees that valid inequalities of the PCKP-polytope can be lifted to obtain facet defining inequalities for the polytope, using a standard sequential lifting procedure. For valid inequalities arising from minimal induced covers and (1,k)-configurations we identify the lower dimensional polytope for which valid inequalities arising from minimal induced covers and (1,k)-configurations are facet defining. We specifically consider valid inequalities arising from minimal induced covers; the variables for which lifting coefficients have yet to be determined are partitioned into two classes. For one of these classes, we establish a relation between the lifting coefficients and the number of components in two related subgraphs of D. Based on this characterization, these lifting coefficients can be seen to be computable in polynomial time. For the second class of lifting coefficients however, we prove that their computation is strongly NP-hard.

A special case of the PCKP which has received considerable attention is the tree knapsack problem, in which the underlying precedence graph is a tree. Aghezzaf, Magnanti and Wolsey [3] for instance, study the polyhedral structure of the problem. Our results allow for more general graph structures and extend their findings. Moreover, our results easily imply that, for the tree knapsack problem, all lifting coefficients can be obtained in polynomial time. In Section 4.4 we consider valid inequalities arising from K-covers. For these valid inequalities standard sequential lifting techniques cannot always be applied. We show that by applying a related lifting procedure facets of the PCKP-polytope can still be obtained. To illustrate the effect of lifted inequalities and their applicability in integer solution procedures, we report our computational results in Section 4.5.

4.2 Notation and Assumptions

Throughout this chapter, the following definitions and notation will turn out to be convenient. For $(i,j) \in A$, item i is called a *predecessor* of item j and item j is called a *successor* of item i. For all $W \subseteq V$, we denote by $F(W) = \{j \in V \setminus W | \exists i_1 \in W : (i_1,j) \in A, \exists i_2 \in W : (j,i_2) \in A\}$ the set of elements in $V \setminus W$ which are both a successor of an element in W and a predecessor of an element in W, by $P(W) = \{j \in V \setminus W | \exists i \in W : (j,i) \in A\} \setminus F(W)$ the set of predecessors of a set W excluding items in W and $F(W)$, by $T(W) = W \cup P(W) \cup F(W)$ the set of predecessors of set W including W, and by $R(W) = V \setminus T(W)$ the set of remaining items (variables). For ease of exposition, $P(\{i\}), T(\{i\})$ and $R(\{i\})$ will be denoted $P(i), T(i)$ and $R(i)$, respectively. Furthermore, for all $W \subseteq V$ define $a(W) = \sum_{i \in W} a_i$. Note that if $(i,j) \in A$ and $(j,k) \in A$, then, by transitivity of the precedence relations, (i,k) can be assumed to be an element of A. Arcs in A induced by transitivity will be omitted in the figures. Moreover, arcs (i,j) are depicted downward.

The following two assumptions can be made without loss of generality.

Assumption 1. The directed graph D is acyclic.

If D contains a cycle, nodes (variables) in this cycle must either all be included in, or all be excluded from the knapsack. Hence, the cycle can be contracted into a single node, with cumulative value and weight coefficient.

Assumption 2. $a(T(i)) \leq b$, for all $i = 1, ..., n$.

This simply implies that for every item i there exists a feasible solution in which it is included in the knapsack. Items violating this assumption can be deleted from the problem instance.

4.3 Minimal Induced Covers and (1,k)-configurations

In the literature on the polyhedral structure of the standard 0–1 knapsack problem, minimal cover inequalities have been investigated (see for instance Balas [10], Balas and Zemel [11]). In order to generalize these concepts to PCKP, we must take into account that if an item i is included in the knapsack, so must all the items in $T(i)$. In Subsection 4.3.1 we therefore consider a straightforward generalization of minimal covers, the so-called minimal *induced* cover (see Park and Park [64]). We show how valid inequalities for the precedence constrained knapsack polytope can be strengthened using standard sequential lifting techniques. Different lifting orders are discussed, and we derive sufficient conditions for classes of valid inequalities under which lifting leads to facet defining inequalities for the PCKP-polytope.

Given a minimal induced cover $C \subseteq V$, we give a combinatorial characterization of the value of the lifting coefficients for the variables in $P(C)$ in Subsection 4.3.2. This leads to the conclusion that these values can be computed in polynomial time. Subsection 4.3.3 shows that the computation of lifting coefficients for variables in $R(C)$ is, in general, strongly NP-hard, but solvable in polynomial time in the special case where the underlying precedence graph is a tree. Finally, Subsection 4.3.4 concludes this section with an illustrative example.

4.3.1 Generic Sequential Lifting

Item $i \in V$ and $j \in V$ are called *incomparable* if both $(i,j) \notin A$ and $(j,i) \notin A$. A set $W \subseteq V$ is called incomparable if the elements in W are pairwise incomparable. Note that if W is incomparable then $F(W) = \emptyset$.

Definition 4.3.1 *$C \subseteq V$ is a minimal induced cover (MIC) if*

- *C is incomparable*
- *$a(T(C)) > b$*
- *$a(T(C)\backslash\{i\}) \leq b$, for all $i \in C$*

In words, a minimal induced cover is a set of incomparable items, which together do not fit in the knapsack, whereas all but one of them do fit in the knapsack together. The above definition follows the work of Boyd [19]. An alternative definition would be to replace $a(T(C)\backslash\{i\}) \leq b$, for all $i \in C$ by $a(T(C\backslash\{i\})) \leq b$, for all $i \in C$. In fact, the latter inequality appears in the definition of minimal induced covers of Park and Park [64]. However, on close inspection it can be verified that the results of Park and Park [64] are derived under conditions for which the two definitions coincide. Since our results are only applicable under the current definition, we follow the original definition of Boyd [19].

For $C \subseteq V$ a MIC, the following inequality is valid:

$$\sum_{i \in C} x_i \leq |C| - 1 \tag{4.5}$$

We refer to this inequality as the *MIC-inequality*.

We define X to be the set of feasible solutions of the PCKP, and $conv(X)$ to be the convex hull of the set X. Furthermore, for any $W_0, W_1 \subseteq V$ such that $W_0 \cap T(W_1) = \emptyset$, we define the subset $X(W^0|W^1) = X \cap \{x \in \{0,1\}^n \mid x_i = 0, \text{ for } i \in W^0, \text{ and } x_i = 1, \text{ for } i \in W^1\}$. For $W \subseteq V$, we denote by e^W the characteristic vector of W, that is, $e_i^W = 1$, if $i \in W$, and $e_i^W = 0$, otherwise.

Proposition 4.3.1 *(see Boyd [19]) The dimension of $conv(X)$ is $|V|$.*

Proof. The vectors $e^{T(i)}$, $i = 1, ..., |V|$ together with the zero vector give $|V| + 1$ affinely independent vectors in $conv(X)$ (note that we use assumption 2 here). ■

Proposition 4.3.2 *Let $C \subseteq V$ be a MIC. Then the inequality (4.5) is facet defining for $conv(X(R(C)|P(C)))$.*

Proof. By Proposition 4.3.1, the dimension of $conv(X(R(C)|P(C)))$ is $|C|$. We specify $|C|$ affinely independent vectors in $conv(X(R(C)|P(C)))$ satisfying (4.5) at equality. Let $\theta^j = e^{T(C)\setminus\{j\}}$, for all $j \in C$. It can easily be checked that the vectors $\theta^j, j \in C$ satisfy the inequality at equality and are affinely independent. ■

Proposition 4.3.2 enables us to lift the variables in $P(C)$ and $R(C)$ into the MIC-inequality using the following technique (see Nemhauser and Wolsey [62]). Let $B^n = \{0,1\}^n$. For some j, suppose $Y \subseteq B^n$, $Y^0 = Y \cap \{x \in B^n | x_j = 0\}$, and $Y^1 = Y \cap \{x \in B^n | x_j = 1\}$. If the inequality

$$\sum_{i \neq j} \alpha_i x_i \leq \alpha_0 \tag{4.6}$$

is facet defining for $conv(Y^1)$ and $Y^0 \neq \emptyset$, then

$$\sum_{i \neq j} \alpha_i x_i + \alpha_j (1 - x_j) \leq \alpha_0 \tag{4.7}$$

is facet defining for $conv(Y)$ if

$$\alpha_j = \alpha_0 - \max_{x \in Y^0} \left\{ \sum_{i \neq j} \alpha_i x_i \right\} \tag{4.8}$$

Similarly, if (4.6) defines a facet for $conv(Y^0)$ and $Y^1 \neq \emptyset$, then

$$\sum_{i \neq j} \alpha_i x_i + \alpha_j x_j \leq \alpha_0 \tag{4.9}$$

defines a facet of $conv(Y)$ if

$$\alpha_j = \alpha_0 - \max_{x \in Y^1} \left\{ \sum_{i \neq j} \alpha_i x_i \right\} \tag{4.10}$$

In order to apply this lifting technique repeatedly to MIC-inequalities, we have to be careful with the order in which the variables are lifted. Otherwise we might, at some point, violate the condition $Y^0 \neq \emptyset$ when applying (4.8) or the condition $Y^1 \neq \emptyset$ when applying (4.10).

Definition 4.3.2 *For $W \subseteq V$, π is called a PFRS-order (predecessors first, remaining variables second) for W if π is a one-to-one mapping $\pi : P(W) \cup R(W) \rightarrow \{1, ..., |P(W) \cup R(W)|\}$ satisfying the following conditions:*

(i) $\pi(i) < \pi(j)$ *if* $i \in P(W), j \in R(W)$

(ii) $\pi(i) < \pi(j)$ *if* $i, j \in P(W), j \in P(i)$ *(reversed topological ordering on $P(W)$)*

(iii) $\pi(i) < \pi(j)$ *if* $i, j \in R(W), i \in P(j)$ *(topological ordering on $R(W)$)*

Note that under assumption 1 such an order always exists. Given a MIC $C \subseteq V$ and a PFRS-order π for C, for all elements $j \in P(C) \cup R(C)$ we define

$$\begin{aligned} p^\pi(j) &= \{i \in P(C) \cup R(C) \mid \pi(i) < \pi(j)\} && \text{(predecessors of } j \text{ in order } \pi) \\ s^\pi(j) &= \{i \in P(C) \cup R(C) \mid \pi(i) > \pi(j)\} && \text{(successors of } j \text{ in order } \pi) \end{aligned}$$

During the lifting process, variables in $P(C) \cup R(C)$ are lifted sequentially, and hence, the lifting problem for a variable $j \in P(C)$ under a PFRS-order π is defined as follows:

Given a MIC $C \subseteq V$ and its associated partially lifted MIC-inequality

$$\sum_{i \in C} x_i + \sum_{i \in P(C) \cap p^\pi(j)} \alpha_i (1 - x_i) \leq |C| - 1 \tag{4.11}$$

which is valid for $conv(X(R(C)|P(C) \cap s^\pi(j)))$, determine α_j, which equals

$$|C| - 1 - \max_{\substack{x \in X(R(C)|P(C) \cap s^\pi(j)): \\ x_j = 0}} \left\{ \sum_{i \in C} x_i + \sum_{i \in P(C) \cap p^\pi(j)} \alpha_i (1 - x_i) \right\} \tag{4.12}$$

Likewise, for a variable $j \in R(C)$ the lifting problem is defined as follows:

Given a MIC $C \subseteq V$ and an inequality

$$\sum_{i \in C} x_i + \sum_{i \in P(C)} \alpha_i (1 - x_i) + \sum_{i \in R(C) \cap p^\pi(j)} \alpha_i x_i \leq |C| - 1 \tag{4.13}$$

which is valid for $conv(X(R(C) \cap s^\pi(j) | \emptyset))$, determine α_j, which equals

$$|C| - 1 - \max_{\substack{x \in (X(R(C) \cap s^\pi(j) | \emptyset)): \\ x_j = 1}} \left\{ \sum_{i \in C} x_i + \sum_{i \in P(C)} \alpha_i (1 - x_i) + \sum_{i \in R(C) \cap p^\pi(j)} \alpha_i x_i \right\} \tag{4.14}$$

Theorem 4.3.1 *Let*

- $C \subseteq V$ *be a MIC with its corresponding valid MIC-inequality (4.5)*
- π *be a PFRS-order for* C.
- *lifting coefficients for variables in* $P(C)$ *be determined according to (4.12)*
- *lifting coefficients for variables in* $R(C)$ *be determined according to (4.14)*

then the resulting inequality

$$\sum_{i \in C} x_i + \sum_{i \in P(C)} \alpha_i (1 - x_i) + \sum_{i \in R(C)} \alpha_i x_i \leq |C| - 1 \tag{4.15}$$

defines a facet of the PCKP-polytope $conv(X)$.

Proof. We construct $|V|$ affinely independent vectors in $conv(X)$, satisfying the inequality at equality. For $j \in C$, let θ^j be defined as in the proof of Proposition 4.3.2. For all $j \in P(C)$, let θ^j be the vector for which the maximum in (4.12) is attained. W.l.o.g. assume that $\theta^j_i = 1$ for all $i \in P(C) \cap s^\pi(j)$, and $\theta^j_i = 0$, for all $i \in R(C)$. Likewise, for all $j \in R(C)$, let θ^j be the vector for which the maximum in (4.14) is attained. W.l.o.g., assume that $\theta^j_i = 0$, for all $i \in R(C) \cap s^\pi(j)$. Then it is easy to verify that the vectors $\theta^j, j \in V$, satisfy the inequality at equality, and moreover, are affinely independent. ■

In order for the lifting procedure that consists of repeatedly solving (4.12) and (4.14) to be applicable, the maximum that is taken in (4.12) and (4.14) has to be well defined. This is not the case, if the subset over which the maximum is taken is empty. We conclude that the procedure is only valid if, at each iteration, the subset is nonempty. In the current framework, of lifting MIC inequalities, this property is ascertained by the ordering conditions stated in Definition (4.3.2). This result, of course, can be generalized to wider classes of valid inequalities, which may even be defined on subsets which need not necessarily be incomparable.

Theorem 4.3.2 *Let $W \subseteq V$, and let $\alpha^T x \leq \alpha_0$, where $\alpha_j = 0$ for $j \notin W$, be a facet defining inequality for $conv(X(R(W)|P(W)))$. Then, lifting the variables in $P(W)$ and $R(W)$ as specified in (4.8) and (4.10) in PFRS-order yields a facet of the PCKP-polytope $conv(X)$.*

Proof. Let $W \subseteq V$. If $W = \emptyset$, then $X(R(W)|P(W)) = X$, no lifting has to be done, and hence the theorem obviously holds. So assume $W \neq \emptyset$. As we are given a facet defining inequality, $X(R(W)|P(W)) \neq \emptyset$, which implies that the vector $e^{P(W)} \in X$ (i.e. $a(P(W)) \leq b$). Now it only remains to prove that at every step of the lifting process the subset on which the maximum in (4.8) and (4.10) is defined is nonempty. Under a PFRS-order π, when lifting $j \in P(W)$, since $a(P(W) \cap s^\pi(j)) \leq a(P(W)) \leq b$, the vector $e^{P(W) \cap s^\pi(j)}$ is in the corresponding subset. When lifting $j \in R(W)$, the vector $e^{T(j)}$ is in the subset at hand. ■

The conditions on the PFRS-order in Definition 4.3.2 are such that at each step of the lifting process the variables which are fixed do not violate the precedence constraints, and the variables which are fixed to one do not violate the knapsack constraint. Instead of considering a PFRS-order in which all elements in $P(W)$ are lifted before elements in $R(W)$, we might also allow for more general lifting orders, in which an element in $R(W)$ can be lifted before all predecessors in $P(W)$ are lifted. The existence of such an order is again guaranteed by assumption 1. Next, we derive necessary and sufficient conditions for which this class of more general orders yields facet defining inequalities.

Definition 4.3.3 *For $W \subseteq V$, π is called a valid order for W if π is a one-to-one mapping $\pi : P(W) \cup R(W) \rightarrow \{1, ..., |P(W) \cup R(W)|\}$ satisfying the following conditions:*

(i) $\pi(i) < \pi(j)$ *if* $i, j \in P(W) \setminus W, j \in P(i)$ *(reversed topological ordering on P(W))*

(ii) $\pi(i) < \pi(j)$ *if* $i, j \in R(W) \setminus W, i \in P(j)$ *(topological ordering on R(W))*

Theorem 4.3.3 *Let $W \subseteq V$, and let $\alpha^T x \leq \alpha_0$, where $\alpha_j = 0$ for $j \notin W$, be a facet defining inequality for $conv(X(R(W)|P(W)))$. Let π be a valid order for W. Let the lifting coefficients of the variables in $P(W)$ and $R(W)$ be determined as in (4.8) and (4.10). Then the resulting inequality is facet defining for $conv(X)$ if and only if $a((P(W) \cap s_j^\pi) \cup T(j)) \leq b$, for each $j \in R(W)$.*

Proof. Let $W \subseteq V$. If $W = \emptyset$, then $X(R(W)|P(W)) = X$, no lifting has to be done, and hence the theorem obviously holds. So assume $W \neq \emptyset$. Using inductive arguments, when lifting a variable x_j we are given a facet defining inequality for the polytope $conv(X(R(W) \cap s_j^\pi | P(W) \cap s_j^\pi))$. If $j \in P(W)$ then by non-emptiness of the above polytope and the definition of a valid order, the subset on which the maximum as in (4.8) is defined is nonempty since the vector $e^{P(W) \cap s^\pi(j)}$ is in the corresponding polytope. If $j \in R(W)$, then the condition on the weights as mentioned in the theorem guarantees that the subset on which the maximum is defined as in (4.10) is nonempty since the vector

$e^{(P(W)\cap s^*(j))\cup T(j)}$ is in the corresponding polytope. Conversely, assume the condition is not satisfied for a $j \in R(W)$. When lifting the corresponding variable, the subset on which the maximum is defined is empty. ■

It is easy to see that conditions (i) and (ii) in Definition 4.3.3 cannot be removed since then immediately the subset on which the maximum is defined in the lifting procedure will become empty. Hence the class of valid orders in Definition 4.3.3 is the most general class of orders which can be considered for the standard lifting procedure as defined by (4.8) and (4.10). Note that the above result paves the way for lifting other classes of valid inequalities. Here, we mention (1,k)-configurations and state the polytope for which the corresponding valid inequality is facet defining.

Definition 4.3.4 *Let*

- $C \cup \{t\} \subseteq V$ *be incomparable, with* $t \notin C$
- $C \cup \{t\}$ *be a cover and* $a(T(C \cup \{t\})\backslash\{t\}) \leq b$
- $Q \cup \{t\}$ *be a minimal (induced) cover,* $\forall Q \subseteq C$ *with* $|Q| = k$ *where* $2 \leq k \leq |C|$

then $C \cup \{t\}$ *is called a (1,k)-configuration.*

For a (1,k)-configuration the following inequalities are valid:

$$(r-k+1)x_t + \sum_{i \in Z(r)} x_i \leq r \tag{4.16}$$

where r is such that $k \leq r \leq |C|$ and $Z(r)$ is any subset of C, with $|Z(r)| = r$.

Proposition 4.3.3 *Let* $C \cup \{t\} \subseteq V$ *be a (1,k)-configuration, and let* $Z(r)$ *be any subset of* C*, with cardinality* $|Z(r)| = r$*. Then it holds that the inequality (4.16) is facet defining for* $conv(X(R(C \cup \{t\}) \cup (C \setminus Z(r))|P(C \cup \{t\})))$.

Proof. It follows similarly as in Proposition 4.3.1 that the dimension of $conv(X(R(C \cup \{t\}) \cup (C \setminus Z(r))|P(C \cup \{t\})))$ is $r+1$. Hence, to prove the proposition, we construct $r+1$ affinely independent vectors in the polytope satisfying the inequality at equality. We assume without loss of generality that the elements are numbered such that $1, ..., r$ denote the elements in $Z(r)$ and $r+1$ denotes element t.

For $i = 1, ..., r-k+2$, let y^i be defined by

$$y^i_j = \begin{cases} 1 & j \in \{i, ..., i+k-2\} \\ 0 & j \in \{1, ..., r\} \setminus \{i, ..., i+k-2\} \\ 1 & j = r+1 \\ 1 & j \in P(C \cup \{t\}) \\ 0 & j \in R(C \cup \{t\}) \cup (C \setminus Z(r)) \end{cases}$$

For $i = r-k+3, ..., r$, let y^i be defined by

$$y^i_j = \begin{cases} 1 & j \in \{i, ..., r\} \cup \{1, ..., i+k-r-2\} \\ 0 & j \in \{i+k-r-1, ..., i-1\} \\ 1 & j = r+1 \\ 1 & j \in P(C \cup \{t\}) \\ 0 & j \in R(C \cup \{t\}) \cup (C \setminus Z(r)) \end{cases}$$

and define y^{r+1} as

$$y^{r+1}_j = \begin{cases} 0 & j = r+1 \\ 1 & j \in Z(r) \\ 1 & j \in P(C \cup \{t\}) \\ 0 & j \in R(C \cup \{t\}) \cup (C \setminus Z(r)) \end{cases}$$

Then it can easily be verified that the vectors $y^i, i = 1, ..., r+1$ are affinely independent and satisfy the inequality at equality. ■

Applying Theorem 4.3.2 or 4.3.3 now yields that the elements of $V \setminus (Z(r) \cup \{t\})$ can now be lifted in PFRS-order or valid order so as to obtain a facet of $conv(X)$.

4.3.2 Lifting Predecessor Variables of a Minimal Induced Cover using a PFRS-order

In general, calculating α_j by (4.8) or (4.10) requires solving a difficult maximization problem. In fact, for PCKP, the optimization problems in (4.12) and (4.14) are in turn PCKP problems. In this Subsection we show that the lifting problem of predecessors under a PFRS-order has a combinatorial interpretation that leads to an algorithm that solves the lifting problem in polynomial time. In contrast, in Subsection 4.3.3, it is shown that the lifting problem for the remaining variables is strongly NP-hard.

Definition 4.3.5 *Let $C \subseteq V$ be a MIC. For $W \subseteq P(C)$ let $f(W)$ be the number of (weak) components in the subgraph of G induced by $W \cup C$.*

Lemma 4.3.1 *f is supermodular: for all $W_1 \subseteq W_2 \subseteq P(C)$ and $i \in P(C) \setminus W_2$ it holds that*

$$f(W_1 \cup \{i\}) - f(W_1) \leq f(W_2 \cup \{i\}) - f(W_2)$$

Proof. Choose $W_2 \subseteq P(C)$ and $W_1 \subseteq W_2$ arbitrarily. Notice first that, since we only consider $W_1 \subseteq P(C)$ and $W_2 \subseteq P(C)$, each component of a subgraph induced by $W_1 \cup C$ or $W_2 \cup C$ contains at least one element $c \in C$. Further, since $W_1 \subseteq W_2$, it must hold that if $c_1 \in C$ and $c_2 \in C$ are in different components of the subgraph induced by $W_2 \cup C$, then they also are in different components of the subgraph induced by $W_1 \cup C$.

Now, consider a component of the graph induced by $W_2 \cup C \cup \{i\}$, containing vertices Q, and let $K = Q \setminus C$. Since this component contains at least one element of C, the subgraph induced by $Q \setminus \{i\}$, consists of a strictly positive number of say k components. Let $K_j, j = 1, \ldots, k$ be the nodes in the intersection of W_2 and the j-th of these components (the components may be numbered arbitrarily), and similarly, let $C_j, j = 1, \ldots, k$ be the nodes in the intersection of C and the j-th of these components. The subgraphs induced by $K_j \cup C_j, j = 1, \ldots, k$ form the distinct components of $W_2 \cup C$.

We first consider the case where $i \in Q$. Since for all $j = 1, \ldots, k$, the subgraphs induced by $K_j \cup C_j$ contain at least one element of C, and since they are distinct components of the subgraph induced by $W_2 \cup C$, there must be distinct $c_j \in C_j, j = 1, \ldots, k$ such that $i \in P(c_j), j = 1, \ldots, k$. Using that $W_1 \subseteq W_2$, we establish that all $c_j, j = 1, \ldots, k$ are in distinct components of the subgraph induced by $W_1 \cup C$. Let K'_j be the set of elements of $P(C)$ that are in the component containing c_j in the subgraph induced by $W_1 \cup C$. Then, $K'_j \subseteq K_j$, since $W_1 \subseteq W_2$. Further, since $i \in P(c_j), j = 1, \ldots, k$, these k components K'_j are in a single component of the subgraph induced by $W_1 \cup C \cup \{i\}$. Let K' be the intersection of the set of nodes in this component and the nodes in $P(C)$, and let C' be the other nodes in the component. Hence, the component Q of the subgraph induced by $(W_2 \cup C \cup \{i\})$ containing i, which consists of k distinct components $K_1 \cup C_1, \ldots, K_k \cup C_k$ of the subgraph induced by $W_2 \cup C$, contains as a subgraph a component $K' \cup C'$ of the subgraph induced by $W_1 \cup C \cup \{i\}$ that contains in turn at least k components $K'_1 \cup C'_1, \ldots, K'_k \cup K'_k$ of the subgraph induced by $W_1 \cup C$, such that $K'_j \subseteq K_j$ for $j = 1, \ldots, k$. Thus it holds that

$$f(W_2 \cap Q) - f((W_2 \cap Q) \cup \{i\}) \geq f(W_1 \cap Q) - f(W_1 \cap Q) \cup \{i\})$$

On the other hand, if $i \notin Q$, $Q \setminus \{i\} = Q$, and hence the subgraph induced by $Q \setminus \{i\}$ consists of one component of the subgraph induced by $W_2 \cup C$, namely the component induced by the vertices in Q. Now consider the set $Q \cap C$, and observe that i is not a predecessor of any of the elements in $Q \cap C$. Consider a component Q' in $W_1 \cup C$ for which it holds that $Q' \subseteq Q$. Since i is not a predecessor of any vertex in Q, it is not a predecessor of any element of Q', and thus, Q' is a component of $W_1 \cup C \cup \{i\}$. Hence, in this case we have that

$$0 = f(W_2 \cap Q) - f((W_2 \cap Q) \cup \{i\}) = f(W_1 \cap Q) - f((W_1 \cap Q) \cup \{i\})$$

Since each component of $W_1 \cup \{i\}$ is contained in a component of $W_2 \cup \{i\}$, and since the components of $W_2 \cup \{i\}$ are, by definition, disjoint, this implies that

$$f(W_2) - f(W_2 \cup \{i\}) \geq f(W_1) - f(W_1) \cup \{i\})$$

as required. ■

Definition 4.3.6 *Let $C \subseteq V$ be a MIC and let $P(C) = \{1, ..., m\}$. Then, for $i \in P(C)$, let γ_i be defined by*

$$\gamma_i = f(\{1, ..., i-1\}) - f(\{1, ..., i\}). \tag{4.17}$$

Thus, γ_i represents the reduction in the number of components by adding node i and the arcs constituting the precedence relations in which i is involved to the subgraph of G induced by $C \cup \{1, \ldots, i-1\}$. As node i is in $P(C)$, this automatically implies that $\gamma_i \geq 0$.

Proposition 4.3.4 *Let $C \subseteq V$ be a MIC and $W \subseteq P(C)$ with $W = \{1, ..., m\}$. Consider the subgraph of G induced by $W \cup C$. For each component K of this subgraph it holds that*

$$\sum_{i \in K \cap W} \gamma_i = |C \cap K| - 1$$

Proof. We use induction on the elements in W. For $i = 1$, consider any component K of the subgraph induced by $C \cup \{i\}$. If component K does not contain node i, the result follows immediately, as both the summation of the coefficients γ_i in K and $|C \cap K| - 1$ equal zero. If component K does contain node i, this node is connected with $|C \cap K|$ nodes in the component K. Hence, $|C \cap K|$ components have been merged into one component, such that the reduction in the number of components γ_i equals $|C \cap K| - 1$.

Next assume that the result holds for the graph induced by $C \cup \{1, ..., i-1\}$. Let K be any component of the graph induced by $C \cup \{1, ..., i\}$. If K does not contain node i, the result follows directly from the induction hypothesis. If K does contain node i, then node i merges a number of components of the graph induced by $C \cup \{1, ..., i-1\}$ together, say components $K_1, ..., K_k$. Note that this implies $\gamma_i = k - 1$. Consequently,

$$\sum_{i \in K \cap \{1, ..., i\}} \gamma_i = |C \cap K_1| - 1 + ... + |C \cap K_k| - 1 + k - 1 = |C \cap K| - 1$$

■

In the sequel we will use a special ordering on the elements from $P(C)$, namely, a reversed topological ordering, i.e. a one-to-one mapping $\pi : P(C) \rightarrow \{1, ..., |P(C)|\}$ satisfying $\pi(i) < \pi(j)$ for $i, j \in P(C)$ and $j \in P(i)$. The following theorem shows that under a reversed topological ordering γ_j is exactly the lifting coefficient α_j, as defined in (4.12), for $j \in P(C)$.

Theorem 4.3.4 *Let $C \subseteq V$ be a MIC and let π be a reversed topological ordering on $P(C)$. If the lifting coefficients γ_i are determined according to (4.17) under the order π, then for each $j = \pi^{-1}(1), ..., \pi^{-1}(|P(C)|)$ the inequality*

$$\sum_{i \in C} x_i + \sum_{i=\pi^{-1}(1)}^{\pi^{-1}(j)} \gamma_i(1 - x_i) \leq |C| - 1 \tag{4.18}$$

is both valid and facet defining for the polytope $conv(X(R(C)|A))$, where the set A is defined as $A = \{\pi^{-1}(j+1), \ldots, \pi^{-1}(|P(C)|)\}$.

Proof. To show validity of (4.18) it suffices to show this for $j = \pi^{-1}(|P(C)|)$. For other values of j validity then follows from the fact that we restrict the set of feasible solutions by setting the variables in $\{\pi^{-1}(j+1), ..., \pi^{-1}(|P(C)|)\}$ to 1. Let x be an arbitrary feasible solution with $x_i = 0$ for all $i \in R(C)$. Define $C' = \{i \in C | x_i = 1\}$, and $W = \{i \in P(C) | x_i = 0\}$. Then obviously $C' \neq C$ since C is a MIC: $|C'| \leq |C| - 1 < |C|$. Hence,

$$\begin{aligned} \sum_{i \in C} x_i + \sum_{i=\pi^{-1}(1)}^{\pi^{-1}(|P(C)|)} \gamma_i(1 - x_i) &= |C'| + \sum_{i \in W} \gamma_i \leq |C'| + \sum_{i \in W} \gamma'_i \\ &\leq |C'| + |C| - f(W) \leq |C| - 1 \end{aligned}$$

The first inequality follows from the supermodularity of f, where the γ_i are obtained using the sequence of $P(C) = \pi^{-1}(1), ..., \pi^{-1}(|P(C)|)$ and the γ'_i are obtained using the subsequence of $\pi^{-1}(1), ..., \pi^{-1}(|P(C)|)$ defined by W. Using (4.17), observe that $\sum_{i \in W} \gamma'_i$ equals $f(\emptyset) - f(W) = |C| - f(W)$. Finally, $f(W)$ equals the number of components in the graph induced by $C \cup W$. If $W = \emptyset$, then $f(W) = |C|$, and the result follows from $C' \subset C$ since C is a MIC. If $W \neq \emptyset$, it follows from solution vector x that nodes in C' are not successors of nodes in W. Hence, in the graph induced by $C \cup W$ the nodes in C' are $|C'|$ individual components. As there is at least one component containing elements in W, it follows that $f(W) \geq |C'| + 1$, which completes the proof of the validity of (4.18).

To show that (4.18) is facet defining, we use induction on the lifting elements j in the order $\pi^{-1}(1), ..., \pi^{-1}(|P(C)|)$. It suffices to display a feasible solution θ^j with $\theta^j_j = 0$, $\theta^j_i = 1$, $i \in \{\pi^{-1}(j+1), \ldots, \pi^{-1}(|P(C)|)\}$, and $\theta^j_i = 0$ for $i \in R(C)$, satisfying (4.18) at equality (see Nemhauser and Wolsey [62] proposition 1.1., page 261). To construct this vector consider the graph induced by $C \cup \{\pi^{-1}(1), ..., \pi^{-1}(j)\}$. Let K be the component containing node j. Define the vector θ^j as follows:

$$\theta^j_i = \begin{cases} 1 & i \in T(C) \backslash K \\ 0 & \text{otherwise} \end{cases}$$

To show that θ^j satisfies (4.18) at equality, note that

$$\begin{aligned}\sum_{i\in C}\theta_i^j + \sum_{i=\pi^{-1}(1)}^{\pi^{-1}(j)}\gamma_i(1-\theta_i^j) &= |C\backslash K| + \sum_{i\in P(C)\cap K}\gamma_i \\ &= |C\backslash K| + |C\cap K| - 1 \\ &= |C| - 1\end{aligned}$$

where the second equality follows from Proposition 4.3.4.

Hence, for $j = \pi^{-1}(1)$ the vector θ^j satisfies the requirements. Next assume that the result holds for all $i \in \{\pi^{-1}(1), ..., \pi^{-1}(j-1)\}$. Hence, we are given $C+j-1$ affinely independent vectors which all have their i^{th} component equal to 1 for $i \in P(C)\backslash\{\pi^{-1}(1), ..., \pi^{-1}(j-1)\}$. By defining θ^j as in the above and following the same reasoning, θ^j satisfies (4.18) at equality and is affinely independent from the foregoing vectors, as $\theta_j^j = 0$. ■

Theorem 4.3.4 shows that f can be used in a lifting process to obtain facet defining inequalities. The coefficients γ_i represent the reduction in the number of components by adding node i and the arcs constituting the precedence relations in which i is involved to the subgraph of G induced by $C\cup\{1,\ldots,i-1\}$. This reduction number can be determined using a set union algorithm such as developed by Tarjan [73]. If n is the number of nodes in the graph, and m is the number of arcs, the algorithm runs in $\mathcal{O}((n+m)\alpha((n+m),n))$, where $\alpha((n+m),n)$ is a functional inverse of Ackerman's function [2].

4.3.3 Lifting Non-Predecessor Variables of a Minimal Induced Cover

Although the maximization problem in (4.8) is NP-hard in general, for PCKP the resulting problem in (4.12) can be solved in polynomial time when lifting predecessors of a MIC C. For variables in $R(C)$, the maximization problem in equality (4.14) is also essentially a PCKP, but will turn out to be strongly NP-hard in general. To give a formal proof of this statement, we introduce the following problem definitions:

Clique (see Garey and Johnson [37])
INSTANCE: Graph $G = (U, E)$, and a positive integer K, with $3 \leq K \leq |U|$.
QUESTION: Does G contain a clique of size K or more?

Note that the assumption $K \geq 3$ does not change the complexity of the problem.

PCKP-MIC-R(C)-lifting
INSTANCE:

- Instance I of $PCKP$, consisting of a directed acyclic graph $D^I = (V^I, A^I)$, a knapsack capacity $B^I \in \mathbb{Z}^+$, and for all $i \in V^I$ a value $c_i^I \in \mathbb{Z}$ and a weight $a_i^I \in \mathbb{Z}^+$.

- A MIC $C^I \subseteq V^I$
- A PFRS-order π^I on $V^I \backslash C^I$
- A facet defining inequality

$$\sum_{i \in C^I} x_i + \sum_{i \in P(C^I)} \alpha_i (1 - x_i) \leq |C^I| - 1$$

obtained by applying the lifting procedure as defined in (4.12) to the variables in $P(C^I)$ under PFRS-order π^I.

- $\alpha^I \in \mathbb{Z}$

QUESTION: Is the lifting coefficient for the first variable in $R(C^I)$ under order π^I, as defined in (4.14), less than or equal to α^I?

Theorem 4.3.5 *PCKP-MIC-R(C)-lifting is NP-complete in the strong sense.*

Proof. It can easily be checked that PCKP-MIC-R(C)-lifting is in NP. Hence it suffices to show that Clique reduces to PCKP-MIC-R(C)-lifting. To this purpose, the graph $G = (U, E)$ will be transformed into an instance $D^I = (V^I, A^I)$, in which there is a node for each $u \in U$ and for each $e \in E$. The nodes corresponding to U will function as predecessors of nodes corresponding to E. As shown in Subsection 4.3.2, a lifting coefficient of a predecessor is equal to the reduction in the number of components. Under the assumption that, for all $u \in U$, the degree $|\delta(u)| \geq 2$, this reduction, and hence the lifting coefficients, are enforced to be 1. Since we are looking for cliques of size at least 3, this assumption causes no loss of generality.

Let $G = (U, E)$ be an instance of Clique, let K be any integer satisfying $3 \leq K \leq |U|$, and let π be any order on the nodes in U, i.e., $U = \{\pi^{-1}(1), ..., \pi^{-1}(n)\}$. We define an index set J, consisting of nodes $u \in U$ which are currently not adjacent to a node with higher index. Hence, $J = \{u \in U | \nexists w \in U : \pi(w) > \pi(u), \{u, w\} \in E\}$. Extend the graph G to $\tilde{G} = (\tilde{U}, \tilde{E})$, where

$$\tilde{U} = U \cup (\bigcup_{u \in J} \{\tilde{u}\}) \qquad \tilde{E} = E \cup (\bigcup_{u \in J} \{u, \tilde{u}\}).$$

Furthermore, let $\tilde{K} = K$, and $\tilde{\pi}$ an order on $\tilde{U}$ such that $\pi(u) = \tilde{\pi}(u)$, for $u \in U$. Then, since nodes in $\tilde{U}$ have degree 1, G contains a clique of size $K \geq 3$ if and only if $\tilde{G}$ contains a clique of size $\tilde{K}$.

Next, we determine a subset $L \subseteq E$ whose elements share a common predecessor in the directed graph to be introduced shortly. Let $\delta(u)$ be the set of edges incident to node u in graph $\tilde{G}$. Then, the set L is determined by the following algorithm, which is to be explained shortly:

```
L = ∅; W = Ē;        /* initialization */
for u = π̃⁻¹(1) to π̃⁻¹(|U|) do
    begin
        let ē ∈ W ∩ δ(u);
        L = L ∪ (W ∩ δ(u)\{ē});
        W = W\δ(u);
    end;
endfor;
```

This algorithm processes the vertices $u \in \tilde{U}$ in increasing order of their indices, and considers the intersection of $\delta(u)$ and W, where W initially consist of all edges in $\tilde{E}$. In each iteration, the algorithm selects an arbitrary edge $\bar{e}$ in the intersection of $\delta(u)$ and W, and eliminates all edges in $\delta(u)$ from W. Except for $\bar{e}$, the thus eliminated edges are added to a set L which is initially empty. Notice that such an $\bar{e}$ always exists since each vertex $u \in U$ is adjacent to a higher index vertex in $\tilde{U}$. Notice also that all edges in $W \cap \delta(u) \setminus \{\bar{e}\}$ are in E since vertices in U only contain edges to $\tilde{U}$, and hence in $\tilde{E} \setminus E$, if they are not adjacent to a higher index vertex in U, and should this be the case, then they contain only one such edge (which per force is chosen to be $\bar{e}$).

Now, we are able to define the instance I with directed graph $D^I = (V^I, A^I)$ where $V^I = \tilde{U} \cup \tilde{E} \cup \{Q, q\}$ and $A^I = \{(u,e) | u \in \tilde{U}, e \in \tilde{E}, e \in \delta(u)\} \cup (Q,q) \cup (\bigcup_{\bar{e} \in L} (Q, \bar{e}))$. To complete the instance I of PCKP-MIC-R(C)-lifting, let $c_v^I \in \mathbb{Z}$ for all $v \in V^I$, and $a_Q^I \in \mathbb{Z}^+$. Further, let $B^I = |U| \cdot \tilde{K}^2 + |\tilde{U} \setminus U| \cdot \left[\tilde{K}^3 + \binom{\tilde{K}}{2}\right] + |\tilde{E}| - 1 + a_Q^I$ and define weights for nodes in V^I as follows:

$$a_u^I = \begin{cases} \tilde{K}^2 & u \in U \\ \tilde{K}^3 + \binom{\tilde{K}}{2} & u \in \tilde{U} \setminus U \\ 1 & u \in \tilde{E} \\ a_Q^I & u = Q \\ B^I - \left[\tilde{K}^3 + \binom{\tilde{K}}{2} + a_Q^I\right] & u = q \end{cases}$$

Now consider $\tilde{E}$. In order to include all $e \in \tilde{E}$ in the knapsack, we must also include all items in $P(\tilde{E})$, which means in this case that all $u \in \tilde{U}$ and Q must be in the knapsack. Since the nodes in $u \in \tilde{U}$ and Q already account for a weight of $|U| \cdot \tilde{K}^2 + |\tilde{U} \setminus U| \cdot \left[\tilde{K}^3 + \binom{\tilde{K}}{2}\right] + a_Q^I$ it follows that only $|\tilde{E}| - 1$ elements from $\tilde{E}$ can be included in the knapsack. This yields that $\tilde{E}$ is a MIC. Now, let π^I be a PFRS-order on $\tilde{U} \cup \{Q, q\}$ defined by:

$$\pi^I(i) = \begin{cases} 1 & i = Q \\ \tilde{\pi}(i) + 1 & i \in \tilde{U} \\ |\tilde{U}| + 2 & i = q \end{cases}$$

and the corresponding facet defining inequality after lifting Q and $\tilde{U}$ be given by

$$\sum_{i\in\tilde{E}} x_i + (|L|-1)(1-x_Q) + \sum_{i\in U}(1-x_i) \leq |\tilde{E}| - 1 \tag{4.19}$$

The lifting coefficients can be explained as follows. Notice that by the construction of A^I, Q connects $|L|$ elements of $\tilde{E}$, and thus adding Q yields a reduction in the number of components of $|L|-1$. Further, by the construction of D^I, there are $|U|$ edges not connected to Q. Now, using our assumption all vertices $u \in U$ have $|\delta(u)| \geq 2$, each vertex $u \in U$ connects the edge e selected in the $\tilde{\pi}(u) - th$ iteration of the previously described algorithm to the component which became connected when x_Q was lifted. Hence $\alpha_u = 1$ for all $u \in U$. Finally, the vertices $\tilde{u} \in \tilde{U}$ can be seen not to cause a reduction in the number of components at all, and hence have coefficient $\alpha_{\tilde{u}} = 0$. Finally, let $\alpha^I = |\tilde{E}| - 1 - \left[\binom{\tilde{K}}{2} + |U| - \tilde{K}.\right]$. We leave it to the reader to verify that the above transformation from G, via $\tilde{G}$, to instance I is polynomial.

Next let us consider the problem (4.14) that arises when lifting x_q, the only variable in $R(C^I)$. When lifting variable x_q according to (4.14), nodes q and Q must be included in the knapsack.

We are now going to show that $\tilde{G}$ contains a clique of size $\tilde{K} \geq 3$ if and only if the maximal value of the lifting coefficient for variable x_q is less than or equal to α^I.

($\Rightarrow$) If G contains a clique of size $\tilde{K}$, then by including the nodes u in $\tilde{U} \cup \tilde{E}$ corresponding to the vertices and edges in the clique in the knapsack together with Q and q, we have a cumulative weight $\tilde{K}^3 + \binom{\tilde{K}}{2} + a^I_Q + B^I - \left[\tilde{K}^3 + \binom{\tilde{K}}{2} + a^I_Q\right] = B^I$. Thus this set of items is feasible. Furthermore, together these items yield a value for α^I of $|\tilde{E}| - 1 - \left[\binom{\tilde{K}}{2} + |U| - \tilde{K}\right]$, as required.

($\Leftarrow$) Let the value of the lifting coefficient for x_q (being $\alpha_q \leq \alpha^I$) be obtained by a solution in which i items (say) corresponding to nodes $u \in U$ in the graph $\tilde{G}$ are included in the knapsack. Let us first assume $i > \tilde{K}$. Then the weight in the knapsack amounts to at least $\tilde{K}^3 + \tilde{K}^2 + a^I_Q + B^I - \left[\tilde{K}^3 + \binom{\tilde{K}}{2} + a^I_Q\right] > B^I$, which yields a contradiction. Hence $i \leq \tilde{K}$. Assume $i < \tilde{K}$. Then, $\alpha_q \geq |\tilde{E}| - 1 - \left[\binom{i}{2} + |U| - i\right] > |\tilde{E}| - 1 - \left[\binom{\tilde{K}}{2} + |U| - \tilde{K}\right] = \alpha^I$, for $\tilde{K} \geq 3$. Again a contradiction. Let us finally consider the case where $i = \tilde{K}$. Then by the above reasoning we find $\alpha_q \geq \alpha^I$ which together with $\alpha_q \leq \alpha^I$ implies that $\alpha_q = \alpha^I$ and hence the remaining knapsack capacity $B^I - \tilde{K}^3 - a^I_Q - B^I + \left[\tilde{K}^3 + \binom{\tilde{K}}{2} + a^I_Q\right] = \binom{\tilde{K}}{2}$ must account for an increase in the solution of the maximization problem in (4.14) of at least $\binom{\tilde{K}}{2}$. Hence, at least $\binom{\tilde{K}}{2}$ vertices from $\tilde{E}$ must be included in the knapsack, and moreover, these vertices must have their predecessors in the graph in the knapsack. This can only be achieved if these vertices are in E, and hence we have found a clique of size $\tilde{K} = K$ in the graph G. ■

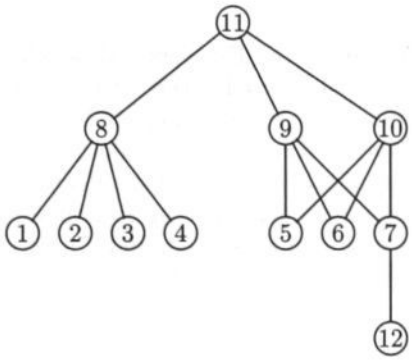
Although lifting variables for a PCKP is strongly NP-hard in general, in the special

case where the precedence graph is a tree and the size of the coefficients in the given valid inequality is polynomially bounded, then lifting coefficients can be obtained in polynomial time.

Theorem 4.3.6 *Given a PCKP for which the precedence graph is a tree, and a valid inequality with coefficients whose size is bounded by a polynomial in the size of the tree, then all lifting coefficients can be determined in polynomial time.*

Proof. Lifting a variable requires solving a tree knapsack problem on a subtree of the original tree. Tree knapsack problems with possibly negative objective coefficients can be solved in pseudo-polynomial time $\mathcal{O}(nQ^2)$ by an extension of a standard dynamic programming algorithm for tree knapsacks (see for instance Johnson and Niemi ([51]), where Q is an upper bound on the maximum value that can be achieved in the optimization problem. If all coefficients of the inequality are polynomially bounded in the size of the original tree, then Q is polynomially bounded, and therefore the tree knapsack problem can be solved in polynomial time. ■

4.3.4 An Example



$a_i = 4, \quad i \in \{1, ..., 7\}$

$a_i = 5, \quad i \in \{8, ..., 11\}$

$a_i = 7, \quad i \in \{12\}$

$b = 37$

Figure 4.1: Example of MIC and (1,k)-configuration

Consider the example in Figure 4.1. Table 4.1 gives an example of both a minimal induced cover for which two different PFRS-orders are stated and a (1,k)-configuration for which only two valid inequalities out of the total set as defined by (4.16) are listed. The resulting facet defining inequalities are listed below. The combinatorial interpretation of the predecessors of a minimal induced cover can be seen from this table, which also illustrates that different PFRS-orders can lead to different facets. The reader may note

that many more covers and (1,k)-configurations are present in the problem instance (see also Section 4.5).

Table 4.1: A minimal induced cover and (1,k)-configuration for problem instance in Figure 4.1.

C	t	type	$Z(r)$	lifting order
{1,2,5,6,7}	-	MIC	-	8,9,10,11,3,4,12
{1,2,5,6,7}	-	MIC	-	8,10,9,11,3,4,12
{1,2,5}	12	(1,2)-conf	{1,2}	7,8,9,10,11,3,4,6
{1,2,5}	12	(1,2)-conf	{1,2,5}	7,8,9,10,11,3,4,6

The resulting facet defining inequalities for Table 4.1 are as follows:

$$\begin{array}{llll}
x_1 + x_2 + x_5 + x_6 + x_7 + (1-x_8) + 2(1-x_9) & & +(1-x_{11}) + x_{12} \leq 4 \\
x_1 + x_2 + x_5 + x_6 + x_7 + (1-x_8) & + 2(1-x_{10}) & +(1-x_{11}) + x_{12} \leq 4 \\
x_1 + x_2 + (1-x_8) & & +(1-x_{11}) + x_{12} \leq 2 \\
x_1 + x_2 + x_5 + (1-x_8) + (1-x_9) & & +(1-x_{11}) + 2x_{12} \leq 3
\end{array}$$

4.4 K-covers

In this section, we discuss valid inequalities arising from K-covers. Although this class of inequalities is a direct generalization of minimal induced covers, it is not always immediately clear for which subset of the polytope the corresponding valid-inequalities are facet defining. One way to obtain facets for the PCKP-polytope would be to follow two steps: first, the exact polytope for which the valid inequality is facet defining could be determined and next, the same lifting procedure as mentioned in Section 4.3 could be applied. In this section, we show that if the first step is skipped, and a different lifting procedure is applied, again facets for the PCKP-polytope are obtained.

Definition 4.4.1 *$C \subseteq V$ is a K-cover if*

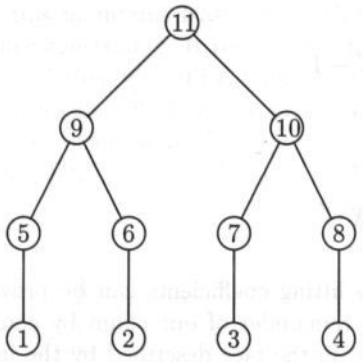
- *C is incomparable*
- *$\forall S \subseteq C$, with $|S| = K$ it holds that $a(T(S)) > b$, but $a(T(S)\backslash\{i\}) \leq b, \forall i \in S$.*

Let $C \subseteq V$ be a K-cover, then PCKP-polytope reads:

$$\sum_{i \in C} x_i \leq K - 1 \tag{4.20}$$

is a valid inequality for the PCKP-polytope.

Figure 4.2 shows that if $C \subseteq V$ is a K-cover the subset $X(P(C)|R(C))$ can be empty, in which case the aforementioned lifting procedure cannot directly be applied to the variables in $P(C) \cup R(C)$. In this example $C = \{1, 2, 3, 4\}$ is a 3-cover, but $X(P(C)|R(C)) = \emptyset$. In fact, the corresponding valid inequality is facet defining for $conv(X(R(C)|\{9, 10, 11\}))$.



$a_i = 2, \quad i \in \{1, 2, 3, 4\}$

$a_i = 7, \quad i \in \{5, 6, 7, 8\}$

$a_i = 1, \quad i \in \{9, 10\}$

$a_i = 3, \quad i \in \{11\}$

$b = 30$

Figure 4.2: Example of K-cover

Below we define a different lifting procedure which is still defined on $P(C) \cup R(C)$ and generates facet defining inequalities. Note that the only difference between the two lifting procedures consists of the polytope over which the maximization problem is defined.

Definition 4.4.2 *Let C be a K-cover and π be a PFRS-order for C which represents the order of lifting variables. Let the lifting coefficient for a variable $j \in P(C)$ be determined by*

$$\alpha_j = K - 1 - \max_{\substack{x \in X: \\ x_j = 0}} \left[\sum_{i \in C} x_i + \sum_{i \in P(C) \cap p^\pi(j)} \alpha_i (1 - x_i) \right] \tag{4.21}$$

and for a variable $j \in R(C)$ by

$$\alpha_j = K - 1 - \max_{\substack{x \in X: \\ x_j = 1}} \left[\sum_{i \in C} x_i + \sum_{i \in P(C)} \alpha_i (1 - x_i) + \sum_{i \in R(C) \cap p^\pi(j)} \alpha_i x_i \right] \tag{4.22}$$

Theorem 4.4.1 *Let*

- $C \subseteq V$ *be a K-cover*
- π *be a PFRS-order for* C
- *the lifting coefficients for variables in* $P(C)$ *be determined according to (4.21)*
- *the lifting coefficients for variables in* $R(C)$ *be determined according to (4.22)*

then the resulting inequality

$$\sum_{i \in C} x_i + \sum_{i \in P(C)} \alpha_i (1 - x_i) + \sum_{i \in R(C)} \alpha_i x_i \leq K - 1 \tag{4.23}$$

a) *is valid, and* $\alpha_i \geq 0$, *for all* $i \in P(C) \cup R(C)$

b) *defines a facet of the PCKP-polytope* $conv(X)$.

Proof. Both validity and nonnegativity of the lifting coefficients can be proved easily using inductive arguments. We will prove the remainder of our claim by constructing $|V| - 1$ linearly independent vectors (directions) in the face described by the inequality. These vectors are constructed as the difference of two vectors, both with the following properties:

(i) the vector satisfies the knapsack constraint;

(ii) the vector satisfies the precedence constraints;

(iii) the vector satisfies the lifted inequality at equality.

Properties (i) and (ii) imply that the vector is in $conv(X)$, whereas property (iii) guarantees that the vector is in the face described by the inequality. For ease of notation, let $P^0(C) = \{j \in P(C) | \alpha_j = 0\}$ and $P^>(C) = \{j \in P(C) | \alpha_j > 0\}$. Note that for each $i \in P^>(C)$ the number of items in C which are *not* successors of i is less than or equal to $K - 2$. If there were more than $K - 2$ items in C which are *not* successors of i, say set $S \subseteq C$ consisting of $K - 1$ items, then the maximization problem in (4.21) for variable x_i would have value at least $K - 1$ since the items in $T(S)$ could be set to one. In other words, if $S \subset C$ contains $K - 1$ elements, then i is a predecessor of at least one element from S. This property is used at several occasions in the remainder of the proof.

Let $C = \{1, ..., |C|\}$. For $j = 1, ..., |C| - 1$ let $C^j \subseteq C \backslash \{j, j+1\}$ with $|C^j| = K - 2$. Next, define

$$\begin{aligned} \phi^j &= e^{T(C^j \cup \{j, j+1\}) \backslash \{j+1\}} \\ \psi^j &= e^{T(C^j \cup \{j, j+1\}) \backslash \{j\}} \\ y^j &= \phi^j - \psi^j \end{aligned}$$

Then the vectors $y^j, j = 1,...,|C|-1$ are clearly linearly independent. Moreover, ϕ^j satisfies properties (i) and (ii) by definition: we take K items from C and all their predecessors; after that we remove one of the items from the K items chosen. Clearly, the vector ϕ^j also satisfies property (iii): we have $K-1$ elements from C, and thus the first term of the left-hand side of (4.23) equals $K-1$. Since the other terms are nonnegative and the equation is valid, we must have that equality holds. For vector ψ^j, (i), (ii), and (iii) can be shown similarly.

For $j \in P^0(C)$, let θ^j be the vector for which the maximum in (4.21) is attained. Let $C^j = \{i \in C | \theta_i^j = 1\}$. Thus, $\forall_{i \in T(C^j)} \theta_i^j = 1$. Moreover, we may assume that $\forall_{i \notin T(C^j)} \theta_i^j = 0$: Clearly, this maintains feasibility with regard to both the knapsack constraint, and the precedence constraints. It remains to show that the maximum in (4.21) is not decreased. If $i \in C \setminus C^j$ then $\theta_i^j = 0$ by definition of C^j. If $i \in P(C) \setminus T(C^j)$, setting $\theta_i^j = 0$ can not have a decreasing effect on the maximum in (4.21), since $(1-x_i)$ becomes positive, and the objective coefficient of $(1-x_i)$ in (4.21) is nonnegative. If $i \in R(C)$, the objective coefficient of x_i is zero in (4.21).

We define

$$\phi_i^j = \begin{cases} 1 & i \in T(C^j) \cup T(j) \cup T(P^>(C) \cap s^\pi(j)) \\ 0 & \text{otherwise} \end{cases}$$

$$\psi_i^j = \begin{cases} 1 & i \in T(C^j) \cup P(j) \cup T(P^>(C) \cap s^\pi(j)) \\ 0 & \text{otherwise} \end{cases}$$

Note that $\phi^j - \psi^j$ is the j-th unit vector, since $j \in T(j)$ but $j \notin T(C^j) \cup P(j) \cup T(P^>(C) \cap s^\pi(j))$ (Note that $\theta_j^j = 0$, see (4.21)). It remains to show that ϕ^j and ψ^j satisfy (i), (ii), and (iii). To show that the knapsack constraint is satisfied, we construct an extension of ϕ that does so. Let $\bar{C}^j$ be an extension of C^j with $K-1$ elements from C. If $j \in T(\bar{C}^j)$ take an arbitrary $i \in C \setminus \bar{C}^j$. Otherwise choose $i \in C \setminus \bar{C}^j$ such that j is a predecessor of i. By definition of K-covers, the set $T(\bar{C}^j \cup \{i\}) \setminus \{i\}$ satisfies the knapsack constraint. Clearly, $T(C^j)$ and $T(j)$ are subsets of $T(\bar{C}^j \cup \{i\}) \setminus \{i\}$. Furthermore, $P^>(C) \subset T(\bar{C}^j)$, since $T(\bar{C}^j)$ contains $K-1$ elements from C. Thus, $P^>(C) \cap s^\pi(j) \subset T(\bar{C}^j)$, and thus $T(P^>(C) \cap s^\pi(j)) \subset T(\bar{C}^j)$, and $T(\bar{C}^j) \subseteq T(\bar{C}^j \cup \{i\}) \setminus \{i\}$.

The precedence constraints hold by construction of ϕ^j.

To show that (iii) is satisfied by ϕ^j, we show that ϕ^j obtains the same value in the maximum of (4.21) as θ^j, i.e., $K-1$. This is true because ϕ^j is an extension of θ^j with elements from $s^\pi(j) \cup \{j\}$. This trivially holds for $P^>(j) \cap s^\pi(j)$ and $\{j\}$. It also holds for the predecessors of both sets, i.e., $T(P^>(j) \cap s^\pi(j))$ and $T(j)$ by the definition of π. Since elements from $s^\pi(j)$ have no contribution to the maximum of (4.21) ϕ^j and θ^j have the same value. Next, the terms in the maximum of (4.21) are a subset of the terms in (4.23) with a value of $K-1$. The remaining terms are nonnegative, and hence, by validity of the inequality, therefore zero. Thus, (4.23) is satisfied at equality. Similar arguments show that (i), (ii), and (iii) hold for ψ^j.

Next, let $j \in P^{>}(C)$, and let θ^j be the vector for which the maximum in (4.21) is attained. W.l.o.g. assume that $\theta^j_i = 0$ for $i \in R(C)$ and for $i \in P(C)$ such that i is not a predecessor of an element in C which is set to one. As indicated in the above, we can again extend the solution θ^j_i to a solution ψ^j, in which all variables in $T(P^{>}(C) \cap s^{\pi}(j))$ are included in the knapsack. Next, let ϕ^j be any vector with $K-1$ elements in C equal to one, and $\phi^j_i = 0$, for $i \in R(C)$. Analogously as in the above, one can verify that ψ^j and ϕ^j satisfy properties (i)-(iii). Define $y^j = \phi^j - \psi^j$, then $y^j_j = 1$ and $y^j_i = 0$, for $i \in P^{>}(C) \cap s^{\pi}(j)$ and $i \in R(C)$.

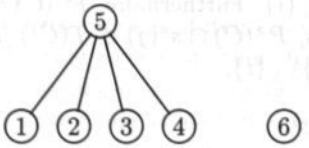
Finally, for $j \in R(C)$, let θ^j be the vector for which the maximum in (4.22) is attained. W.l.o.g., assume that $\theta^j_i = 0$, for $i \in R(C) \cap s^{\pi}(j)$. Let $C^j \subset C$ with $|C^j| = K-1$. Next, define

$$\begin{aligned} \phi^j &= \theta^j \\ \psi^j &= e^{T(C^j)} \\ y^j &= \phi^j - \psi^j \end{aligned}$$

Then both ϕ^j and ψ^j satisfy properties (i)-(iii). Finally, $y^j_j = 1$ and $y^j_i = 0$ for $i \in s^{\pi}(j)$.

We leave it to the reader to verify that the $|V|-1$ vectors as defined in the above are linearly independent. ■

Example 4.4.1 *This example shows that Theorem 4.4.1 does not necessarily hold if we allow for more general lifting orders.*



$$a_i = 4, \quad i \in \{1, ..., 4\}$$

$$a_i = 2, \quad i \in \{5\}$$

$$a_i = 10, \quad i \in \{6\}$$

$$b = 10$$

Figure 4.3: Counterexample

Now the set $\{1,2,3,4\}$ *is a* 3*-cover and hence* $\sum_{i=1}^{4} x_i \leq 2$ *is a valid inequality. If we first lift variable* x_6 *according to (4.22) and then* x_5 *according to (4.21) we obtain the inequality* $\sum_{i=1}^{4} x_i + 2x_6 \leq 2$ *which is valid but not facet defining for* $conv(X)$.

4.5 Computational Results and Concluding Remarks

To gain insight in the effectiveness of the proposed facets in this chapter, reconsider the example of Subsection 4.3.4. In this small example 36 covers were found, which by applying different PFRS-orders led to a total of 63 different facet defining inequalities. Furthermore seven K-covers were found (11 different facets), and 24 (1,k)-configurations (leading to 100 different facets).

In the five problem instances in Table 4.2 nodes in the same "layer" of the graph are given the same, but randomly chosen, objective function coefficient. For each of these 5 problem instances, the value of the LP-relaxation, the IP-value and the value of the LP-relaxation after adding all 174 facets to the description were computed.

Table 4.2: Computational results for instance in Figure 4.1.

problem	LP-value	LP + facets	IP-value	%gap closed
obj1	49.3	47.8	46	45%
obj1a	57.0	51.8	48	58%
obj1b	104.0	104.0	100	0%
obj1c	242.0	235.9	227	41%
obj1d	270.6	244.6	225	57%

These results indicate that the effect of the valid inequalities may be significant. Problem instance obj1b represents a situation where nodes in layer three (i.e. nodes 1 through 7) have a low objective coefficient compared to the other nodes.

Next, 9 objective functions in which each node is given a random objective coefficient were generated. Different ranges of objective coefficients were tested. Table 4.3 reports on the computational results for these problem instances.

Table 4.3: Additional computational results for instance in Figure 4.1.

problem	LP-value	LP + facets	IP-value	%gap closed
obj2	124.0	119.1	119	97%
obj3	116.0	110.3	110	95%
obj4	104.6	97.0	97	100%
obj5	179.8	176.0	176	100%
obj6	231.7	228.3	226	60%
obj7	233.7	223.7	222	85%
obj8	274.0	251.4	248	87%
obj9	53.5	51.5	51	80%
obj10	43.2	41.0	41	100%

The results show that a large proportion of the gap *can* be closed by including the facets

proposed in this chapter. In fact, only PFRS-orders were considered in the tests, hence more valid inequalities can be included by allowing for more general lifting orders as indicated in this chapter. Finally we state some remarks on possible future research directions.

Firstly, note that problem instance obj1b shows that for certain problem instances the facet defining inequalities discussed in this chapter are not very useful. Direct generalizations of other well-known classes of valid inequalities for regular knapsack problems could of course form a fruitful area for future research. Next, to incorporate such inequalities into a branch-and-cut procedure, the separation problem must be addressed. Again, generalizations of separation heuristics for ordinary cover inequalities can be investigated. Thirdly, the difference in the definition of a minimal induced cover used in this chapter and by Boyd ([19]), and the definition employed by Park and Park ([64]) deserves more research. Finally, the lifting procedure as proposed in Section 4.4 is only proven to be a valid procedure for valid inequalities arising from K-covers. In fact, in the proof detailed information from the definition of a K-cover is used. The question arises whether these lifting ideas can also be used for different and/or more general classes of valid inequalities.

Part II

Network Loading Problems

Chapter 5

Models and Algorithms for Network Loading Problems

5.1 Introduction

Many network design problems share the following general problem description. Consider a telecommunication network and a set of telecommunication traffic demands (referred to as commodities) which must be routed between designated nodes in the network. To enable traffic flow through the network sufficient capacity should be installed on the edges of the network. Given routing and capacity installation cost functions, the goal is to design a minimum cost network, that is to design routing schemes for the commodities and capacity installations on the edges, such that all traffic demands can be routed from origin to destination simultaneously. *Network loading problems* (NLP) form a subset of such network design problems and are characterized by two specific properties (cf. [54], which also contains a description of network loading problems). Firstly, a set T of capacity types is available for capacity installation on the edges of the network, and for each type one can install an *integral* number of units on an edge. Secondly, if $\lambda^1 < \ldots < \lambda^{|T|}$ represent the capacity amounts per unit for the different capacity types, then these amounts are *modular*, i.e. $\lambda^{\tau+1}$ is a multiple of λ^τ. Consequently, the amount of capacity installed on an edge of the network is limited to a discrete set of possibilities. Dedicated studies related to these two properties include Pochet and Wolsey [67], and Gabrel and Minoux [35], [36].

Apart from this general problem description, each individual application has its specific characteristics. In some applications traffic flow from individual commodities can be bifurcated on several paths through the network (see for instance Magnanti, Mirchandani and Vachani [55], [56]), whereas other applications require the demand of a commodity to be routed on a single path (see Gavish and Altinkemer [40]). Sometimes the routing of a commodity is even further restricted to a set of so-called hub nodes (see Brockmüller, Günlück and Wolsey [20]). Most of these models assume that capacity installation on an edge is undirected, that is, the capacity on an edge should be greater than or equal to the *sum* of the flow in both directions. On the other hand, Bienstock and Günlück [16] and

Bienstock et al. [15] discuss directed capacity problems, where the capacity on an edge should be greater than or equal to the *maximum* of the flow in both directions. Although many models use flow variables on individual edges to represent the flow of a commodity through the network, other authors employ path variables (see for instance Clarke and Gong [25]). Related work can be found in Dahl, Martin and Stoer [28].

Due to the large traffic flows in telecommunication networks, especially in higher layers of the network hierarchy, the possible breakdown of a network component may have a significant impact on a network's performance. Therefore, telecommunication companies often want to design networks that can cope with the breakdown of critical components. For instance, the implementation of alternative routing schemes for commodities could make the network (partially) insensitive for network component failures. Alternative routing schemes can only be made available if multiple disjoint paths exist between nodes in the graph. We refer to Monma and Shallcross [61], Monma, Munson and Pulleyblank [60], Grotschel and Monma [42] Grotschel, Monma and Stoer [43], [44] for design issues that require a certain degree of connectivity between pairs of nodes in the network. But even if the underlying graph supports multiple routing strategies by the existence of multiple paths between pairs of nodes, sufficient capacity should be available in the network in order to cope with network component failures. Usually, only *single* simultaneous component breakdowns are considered since these are most common. Hence, the protection against multiple simultaneous failures in the network is considered to be too costly. As for the basic network loading problems mentioned in the above, reliability of a network can be defined in many ways (see for instance Alevras, Grotschel and Wessaly [4], Balakrishnan et al [7]). Some applications require the network to be insensitive for edge failures (see Lisser, Sarkissian and Vial [53]) whereas other applications require the network to be insensitive for node failures (see Amiri and Pirkul [5]). Dahl and Stoer [29] consider models where the amount of demand of a single commodity which flows through a network component is restricted, in order to limit the immediate loss of information in case of a network component breakdown. Related models can be found in Paul et al [65], Bienstock and Muratore [17], Bienstock and Saniee [18].

This chapter reports on a joint research project with KPN Research, Leidschendam, the Netherlands. The objective of the project is to gain insight in the various alternatives that are available for non-bifurcated network design, and to design a decision support tool that enables network planners to analyze these distinct network designs. Several specific research topics are defined. The first major topic involves the optimal network layout, that is routing strategies and capacity installation, for non-bifurcated network problems in case the network is not required to cope with failures. Moreover, we are interested in the influence of additional side restrictions which may be imposed by network planners on the network layout and its costs. Therefore, we compare *arbitrary* routing strategies (for which traffic flows can be routed on any path through the network) with so-called *symmetric* routing strategies, for which symmetric traffic flows (that is, pairs of traffic flows for which the origin of the first traffic flow is the destination of the second traffic flow, and vice versa) are routed on the reversed path through the network. Next, routing a traffic flow from its origin to destination via a large number of intermediate nodes could cause a larger delay for traffic flows, which in general is considered to be undesirable. Moreover, due to the larger number of traffic flows that pass through a node in such

routing strategies one could also expect that in case of a network component breakdown a larger set of traffic flows will be affected. Hence, we study the influence of a restriction on the number of nodes on the paths used for routing strategies.

The second major topic of our study involves reliability issues. More precisely we are interested in the design of networks which are capable of routing traffic demands both in fully operational state and in case of a single node breakdown. This can be achieved by using both primary routing schemes (to be used if no node failure occurs) and secondary routing schemes (used if primary routing is not feasible due to node failure), where the two routing schemes are required to be node-disjoint. Again, several questions arise. On the one hand, one could employ two separate networks for primary and secondary traffic flows. On the other hand, perhaps cost reductions could be achieved by integrating these traffic flows into a single network.

The above reliability description is based on a single secondary routing scheme for each commodity, which is to be used in case the primary path is no longer feasible in case of a node failure. Instead, one could also allow for a more general secondary routing scheme where the secondary path for a commodity may depend on the actual node failure. Again, one is interested in the differences in network layout and the impact on costs for the different scenarios.

The third goal of the research project is to incorporate the acquired knowledge and algorithms for the various network loading problems into a decision support system. The primary reason for building a software tool with a user-friendly interface is to make algorithms more accessible to network planners, which allows for more intensive use of the ideas presented in this chapter. Secondly, we aim to reduce the existing gap between theory and practice. In theory, one is often interested in the optimal solution for a mathematical description of a real-life problem, where optimal is usually defined in terms of a single objective function. In practice, network planners want to analyze several network scenarios and solution alternatives, and make a comparison on the basis of multiple criteria. Hence, to efficiently exploit theoretical knowledge in the practical network planning process one needs both easy problem input facilities and algorithm invocation, as well as efficient access to network design solutions. Therefore, we describe some important characteristics of a software tool which supports these requirements.

In Section 5.2 we state several models for basic network loading problems without reliability considerations, whereas network loading problems with reliability constraints are the subject of Section 5.3. Given this set of problems and the research questions as mentioned in the above, the choice of research methodology is motivated in Section 5.4. Section 5.5 reports on a decision support tool which facilitates the network design process for network planners using the developed algorithms. Finally, in Section 5.6 we report on a computational study and address the research questions as posed in the above.

5.2 Network Loading Problems without Reliability Constraints

Let $G = (V, E)$ be an undirected connected graph with node set V and edge set E, and let A be the arc set which is obtained by replacing each edge $e = \{i, j\} \in E$ by two directed arcs (i, j) and (j, i). Let Q be a set of commodities, where each $q \in Q$ is a triple (s^q, t^q, d^q) representing a telecommunication demand of size $d^q \in \mathbb{Q}^+$ that must be routed through the graph on a single path from origin (or source) $s^q \in V$ to destination (or sink) $t^q \in V$. In order to enable traffic in the network sufficient capacity must be installed on the edges of the network. Different types of capacity are available, but only an integral amount of capacity units of each type can be installed on an edge of the network. Let T represent the set of available capacity types, c_{ij}^τ denote the costs per installed unit capacity of type $\tau \in T$ on edge $\{i, j\}$, and let $\lambda^\tau \in \mathbb{Q}^+$ be the associated base capacity per unit for type $\tau \in T$. The goal is to minimize the costs of installed capacity in the network under the restriction that a feasible simultaneous routing for all commodities exists. To formulate this problem as an integer program, define an integer variable $x_{ij}^\tau \in \mathbb{N}$ to measure the number of capacity units of type τ installed on edge $\{i, j\}$, and let f_{ij}^q be a binary variable that indicates whether the commodity $q \in Q$ is routed on arc $(i, j) \in A$ or not. The model then reads :

$$\text{min} \quad \sum_{\{i,j\} \in E} \sum_{\tau \in T} c_{ij}^\tau x_{ij}^\tau \tag{5.1}$$

$$\text{s.t.} \quad \sum_j f_{ij}^q - \sum_j f_{ji}^q = \begin{cases} 1 & i = s^q \\ -1 & i = t^q \\ 0 & i \neq s^q, t^q \end{cases} \quad \forall q \in Q, i \in V \tag{5.2}$$

$$\sum_{q \in Q} d^q (f_{ij}^q + f_{ji}^q) \leq \sum_{\tau \in T} \lambda^\tau x_{ij}^\tau \quad \forall \{i, j\} \in E \tag{5.3}$$

$$f_{ij}^q, f_{ji}^q \in \{0, 1\}, \; x_{ij}^\tau \in \mathbb{N} \quad \forall q \in Q, \{i, j\} \in E, \tau \in T \tag{5.4}$$

The objective function (5.1) measures the total cost of capacity installation on the edges of the network. The flow balance constraints (5.2) guarantee that the demand of a commodity q is routed on a single path from origin s^q to destination t^q. The edge capacity constraints (5.3) imply that sufficient capacity is installed to accommodate the resulting flow on the edges of the network. This model is called the undirected non-bifurcated flow model $UNFM$. It is called *undirected* since capacity on an edge is undirected, that is capacity can be used for traffic in both directions on the edge. The model is called *non-bifurcated* as the demand of a commodity has to be routed on a single path (i.e. the demand cannot be bifurcated). Finally, flow variables on individual arcs are used to model the flow of a commodity from origin node to destination node. Brockmüller, Günlück and Wolsey [20], [21] employ a polyhedral approach to tackle these problems. Amiri and Pirkul [6] consider nonlinear versions of the problem incorporating queueing and delay costs and use Lagrangean techniques. The bifurcated version of the problem is obtained if the flow variables are not required to be integral, consequently, flow can be split among several paths. This bifurcated version is studied by Barahona [12] and Magnanti, Mirchandani and Vachani [55], [56].

Instead of using flow variables on individual edges to model routing restrictions, one can also use path variables to represent the flow of telecommunication traffic through the network. Although in general such models can have a large (exponential) number of variables, the advantage is that certain undesirable paths can easily be excluded from the formulation. For instance, network planners often consider long paths to be unattractive since this might lead to unacceptable delays. Path formulations are considered in for instance Gavish and Altinkemer [40] and Pirkul and Narasimhan [66]. Let Y^q denote the total set of available paths for commodity q, and let the binary variables y_p^q represent whether a certain path $p \in Y^q$ is used to route the commodity q from source node s^q to sink node t^q. If $Y_e^q \subseteq Y^q$ denotes the set of paths for commodity q that contain arc (i,j) or arc (j,i) (where $e = \{i,j\}$), then this leads to the following undirected non-bifurcated path model $(UNPM)$:

$$\begin{aligned}
\min \quad & \textstyle\sum_{\{i,j\}\in E} \sum_{\tau\in T} c_{ij}^{\tau} x_{ij}^{\tau} && && (5.5)\\
\text{s.t.} \quad & \textstyle\sum_{p\in Y^q} y_p^q = 1 && \forall q \in Q && (5.6)\\
& \textstyle\sum_{q\in Q} \sum_{p\in Y_e^q} d^q y_p^q \le \sum_{\tau\in T} \lambda^{\tau} x_{ij}^{\tau} && \forall e = \{i,j\} \in E && (5.7)\\
& y_p^q \in \{0,1\},\ x_{ij}^{\tau} \in \mathbb{N} && \forall q \in Q, p \in Y^q, \{i,j\} \in E, \tau \in T && (5.8)
\end{aligned}$$

Depending on the exact application, capacity on edges in the network can also be directed, i.e. each unit of capacity of type τ installed on an edge $\{i,j\}$ gives a capacity of λ^{τ} on both corresponding arcs (i,j) and (j,i), and capacity consumption is directed as well. This leads to the directed non-bifurcated flow model $(DNFM)$.

$$\begin{aligned}
\min \quad & \textstyle\sum_{\{i,j\}\in E} \sum_{\tau\in T} c_{ij}^{\tau} x_{ij}^{\tau} && && (5.9)\\
\text{s.t.} \quad & \textstyle\sum_j f_{ij}^q - \sum_j f_{ji}^q = \begin{cases} 1 & i = s^q \\ -1 & i = t^q \\ 0 & i \neq s^q, t^q \end{cases} && \forall q \in Q, i \in V && (5.10)\\
& \textstyle\sum_{q\in Q} d^q f_{ij}^q \le \sum_{\tau\in T} \lambda^{\tau} x_{ij}^{\tau} && \forall \{i,j\} \in E && (5.11)\\
& \textstyle\sum_{q\in Q} d^q f_{ji}^q \le \sum_{\tau\in T} \lambda^{\tau} x_{ij}^{\tau} && \forall \{i,j\} \in E && (5.12)\\
& f_{ij}^q, f_{ji}^q \in \{0,1\},\ x_{ij}^{\tau} \in \mathbb{N} && \forall q \in Q, \{i,j\} \in E, \tau \in T && (5.13)
\end{aligned}$$

The bifurcated version of $DNFM$ has been studied in Bienstock and Günlück [16] and Bienstock et al. [15]. Similar as for the undirected case, one can model the directed case using path variables. If $Y_{ij}^q \subseteq Y^q$ denotes the set of paths for commodity q which contain arc (i,j), then the directed non-bifurcated path model $(DNPM)$ reads:

$$\begin{aligned}
\min \quad & \textstyle\sum_{\{i,j\}\in E} \sum_{\tau\in T} c_{ij}^{\tau} x_{ij}^{\tau} && && (5.14)\\
\text{s.t.} \quad & \textstyle\sum_{p\in Y^q} y_p^q = 1 && \forall q \in Q && (5.15)\\
& \textstyle\sum_{q\in Q} \sum_{p\in Y_{ij}^q} d^q y_p^q \le \sum_{\tau\in T} \lambda^{\tau} x_{ij}^{\tau} && \forall \{i,j\} \in E && (5.16)\\
& \textstyle\sum_{q\in Q} \sum_{p\in Y_{ji}^q} d^q y_p^q \le \sum_{\tau\in T} \lambda^{\tau} x_{ij}^{\tau} && \forall \{i,j\} \in E && (5.17)\\
& y_p^q \in \{0,1\},\ x_{ij}^{\tau} \in \mathbb{N} && \forall q \in Q, p \in Y^q, \{i,j\} \in E, \tau \in T && (5.18)
\end{aligned}$$

One of the research issues mentioned in the introductory section is the comparison of arbitrary versus symmetric routing. Here, we state a definition and the equalities that can be added to the models to impose the symmetric routing restriction.

Definition 5.2.1 *Let $q^1, q^2 \in Q$, then (q^1, q^2) is a symmetric commodity pair if $s^{q^1} = t^{q^2}$ and $s^{q^2} = t^{q^1}$. Moreover, $Q^S \subseteq Q \times Q$ denotes the set of all symmetric commodity pairs.*

Given the set of symmetric commodity pairs Q^S, the restrictions $f_{ij}^{q^1} = f_{ji}^{q^2}$ for all $(q^1, q^2) \in Q^S$, for all $\{i, j\} \in A$ can be added to the flow formulation. Indeed, these equalities imply that symmetric commodity pairs are routed on reversed paths, and they also imply that the number of commodities (hence the number of variables) used in the formulation can be decreased. In the path formulations, if $(q^1, q^2) \in Q^S$ one would require that for each path $p^1 \in P^{q^1}$ there exists a reversed path $p^2 \in P^{q^2}$ (and vice versa). Similarly, the restrictions $p^1 = p^2$ can be added to the formulation for such reversed paths and again, the number commodities (variables) can be decreased.

Finally, we address the upper bound on the number of nodes or arcs on the selected path for a commodity. If ℓ denotes the maximum number of nodes on a path from the source to the sink of a commodity (endpoints included, hence $2 \leq \ell \leq |V|$), then in the flow formulation we can add the restriction $\sum_{\{i,j\} \in A} f_{ij}^q \leq (\ell - 1)$, for all $q \in Q$. In the path formulation we simply impose that all paths in the set P^q contain at most ℓ nodes.

5.3 Network Loading Problems with Reliability Constraints

In this section we consider non-bifurcated network loading problems with reliability constraints, which guarantee that in the case of a node breakdown in the network, all commodities (except the commodities for which the failure node is the source or sink node) can still be routed through the network from origin to destination. We consider two types of models. Subsection 5.3.1 discusses reliable network loading models with a primary and a single secondary path for each commodity. When the network is in fully operational state, the primary path is chosen for each commodity. In case of a node failure some of these primary paths are no longer feasible. We say that a commodity is *lost* by a node failure on node k if the source node or sink node of the commodity is node k. Moreover, a commodity is said to be *affected* if it is not lost, but the failure node is on the primary path for the commodity. In case of a node failure, all affected commodities are routed via their secondary path.

Instead of having a single secondary path, one could also employ a set of secondary paths for a commodity, such that in case a commodity is affected by a node failure, the selected secondary path is dependent on the actual failure node. In Subsection 5.3.2 we therefore discuss models that allow for node-dependent secondary paths. Note that the models that are described in these subsections are only used to communicate the problems. Efficiency

or computational tractability, and the fact that multiple models can be employed for reliability models is therefore not an issue in this section.

5.3.1 Network Loading Models with Single Secondary Paths

Here, we choose two node-disjoint paths in the network for each commodity, which are referred to as primary and secondary path. These paths are required to be node disjoint, except for the first node (source) and and last node (sink) on the path for the commodity. If the network is completely intact, the primary path is chosen for each commodity. Let K denote the set of nodes that are subject to failure and for which we want to incorporate reliability measures. In case of a failure of node $k \in K$, affected commodities will be routed via their secondary path. Let Y^q denote the set of all possible primary paths for commodity $q \in Q$, and $Y^q_{ij} \subseteq Y^q$ denote the set of paths that contain the arc (i,j). Similarly, let Z^q be the set of all possible secondary paths for the commodity, and $Z^q_{ij} \subseteq Z^q$ the set of all paths that contain the arc (i,j). For all $p \in \cup_{q \in Q}(Y^q \cup Z^q)$, for all $k \in K$ define a parameter

$$\Delta^k_p = \begin{cases} 1 & \text{if node } k \text{ is on path } p \\ 0 & \text{otherwise} \end{cases}$$

We introduce the following additional decision variables:

$$y^q_p = \begin{cases} 1 & \text{if commodity } q \text{ uses primary path } p \in Y^q \\ 0 & \text{otherwise} \end{cases}$$

$$z^q_p = \begin{cases} 1 & \text{if commodity } q \text{ uses secondary path } p \in Z^q \\ 0 & \text{otherwise} \end{cases}$$

If a node $k \in K$ fails, commodities with origin or destination node k will be lost. Let Q^k denote the set of commodities with both source and sink node not equal to node k, hence Q^k denotes the set of commodities that are not lost in failure state k. Moreover, only affected traffic (i.e. traffic for which node k is on the primary path, but not as source node or sink node) will be rerouted via the secondary path. Therefore, in case of such a failure, an edge in the network will possibly accommodate both flow originating from primary paths, and flow originating from secondary paths. To be able to measure the total flow on an edge in case of a failure at node k, we introduce the following decision variables for all $k \in K, q \in Q^k, p \in Z^q$:

$$r^{qk}_p = \begin{cases} 1 & \text{if commodity } q \text{ uses secondary path } p \text{ and } k \text{ is on the primary path} \\ 0 & \text{otherwise} \end{cases}$$

In words, in case of a failure at node k, the variable r^{qk}_p is used to measure the flow originating from secondary paths. Next, we state the directed version of the problem, the

undirected version is similar.

$$\min \quad \sum_{\{i,j\}\in E}\sum_{\tau\in T} c_{ij}^{\tau} x_{ij}^{\tau} \tag{5.19}$$

$$\text{s.t.} \quad \sum_{p\in Y^q} y_p^q = 1 \qquad \forall q \in Q \tag{5.20}$$

$$\sum_{p\in Z^q} z_p^q = 1 \qquad \forall q \in Q \tag{5.21}$$

$$\sum_{p\in Y^q} \Delta_p^k y_p^q + \sum_{p\in Z^q} \Delta_p^k z_p^q \le 1 \qquad \forall k \in K, q \in Q^k \tag{5.22}$$

$$\sum_{q\in Q}\sum_{p\in Y_{ij}^q} d^q y_p^q \le \sum_{\tau\in T} \lambda^{\tau} x_{ij}^{\tau} \qquad \forall \{i,j\} \in E \tag{5.23}$$

$$\sum_{q\in Q}\sum_{p\in Y_{ji}^q} d^q y_p^q \le \sum_{\tau\in T} \lambda^{\tau} x_{ij}^{\tau} \qquad \forall \{i,j\} \in E \tag{5.24}$$

$$r_p^{qk} \ge z_p^q + \sum_{p\in Y^q} \Delta_p^k y_p^q - 1 \qquad \forall k \in K, \forall q \in Q^k, \forall p \in Z^q \tag{5.25}$$

$$r_p^{qk} \le z_p^q \qquad \forall k \in K, \forall q \in Q^k, \forall p \in Z^q \tag{5.26}$$

$$r_p^{qk} \le \sum_{p\in Y^q} \Delta_p^k y_p^q \qquad \forall k \in K, \forall q \in Q^k, \forall p \in Z^q \tag{5.27}$$

$$\sum_{q\in Q^k} d^q \Big(\sum_{p\in Y_{ij}^q} (1-\Delta_p^k) y_p^q + \sum_{p\in Z_{ij}^q} r_p^{qk}\Big) \le \sum_{\tau\in T} \lambda^{\tau} x_{ij}^{\tau}$$
$$\forall \{i,j\} \in E, \forall k \in K : k \ne i,j \tag{5.28}$$

$$\sum_{q\in Q^k} d^q \Big(\sum_{p\in Y_{ji}^q} (1-\Delta_p^k) y_p^q + \sum_{p\in Z_{ji}^q} r_p^{qk}\Big) \le \sum_{\tau\in T} \lambda^{\tau} x_{ij}^{\tau}$$
$$\forall \{i,j\} \in E, \forall k \in K : k \ne i,j \tag{5.29}$$

$$y_p^q, z_p^q \in \{0,1\}, r_p^{qk} \in [0,1] \qquad \forall q \in Q, \forall p \in Y^q \cup Z^q, \forall k \in K \tag{5.30}$$

$$x_{ij}^{\tau} \in \mathbb{N} \qquad \forall \{i,j\} \in E, \forall \tau \in T \tag{5.31}$$

Constraints (5.20)-(5.21) state that for each commodity one should choose both a primary and a secondary path. The primary and secondary path are enforced to be node-disjoint (except for source and sink node) by constraint (5.22). If no failure occurs, then the capacity on all the edges should be sufficient to accommodate all primary flow, as stated in restrictions (5.23)-(5.24). Constraints (5.25)-(5.27) set the variable r_p^{qk} equal to one if and only if two conditions are satisfied: secondary path p is used and node k is on the primary path. Note that these constraints also imply that, although the variables are binary variables, they can be relaxed to $[0,1]$ variables, since the integrality will be enforced by the model. Finally, restrictions (5.28)-(5.29) guarantee that sufficient capacity is available on the edges in the network in case a failure occurs at node k.

5.3.2 Network Loading Models with Node Dependent Secondary Paths

In this section we assume that the secondary path that a commodity will use if it is affected by a failure at node k may depend on the failure node k. Let K denote the set of possible network states, where $k^0 \in K$ represents the fully operational state, i.e. no node failure occurs, and $k \in K \setminus \{k^0\}$ represents a failure of node k. Let $Y^{qk}(Y_{ij}^{pq})$ denote the set of all feasible paths for commodity q in case of failure state k (that contain arc (i,j)).

For all $k \in K, q \in Q^k, p \in Y^{qk}$, define

$$y_p^{qk} = \begin{cases} 1 & \text{if commodity } q \text{ uses path } p \text{ in network state } k \\ 0 & \text{otherwise} \end{cases}$$

The directed version of the model then reads:

$$\begin{aligned}
\min \quad & \textstyle\sum_{\{i,j\}\in E} \sum_{\tau\in T} c_{ij}^{\tau} x_{ij}^{\tau} && && (5.32)\\
\text{s.t.} \quad & \textstyle\sum_{p\in Y^{qk}} y_p^{qk} = 1 && \forall k \in K, \forall q \in Q^k && (5.33)\\
& \textstyle\sum_{q\in Q^k} \sum_{p\in Y_{ij}^{qk}} d^q y_p^{qk} \le \sum_{\tau\in T} \lambda^{\tau} x_{ij}^{\tau} && \forall k \in K, \forall \{i,j\} \in E : k \neq i,j && (5.34)\\
& \textstyle\sum_{q\in Q^k} \sum_{p\in Y_{ji}^{qk}} d^q y_p^{qk} \le \sum_{\tau\in T} \lambda^{\tau} x_{ij}^{\tau} && \forall k \in K, \forall \{i,j\} \in E : k \neq i,j && (5.35)\\
& y_p^{qk} \ge y_p^{qk^0} && \forall k \in K \setminus \{k^0\}, \forall q \in Q^k, \forall p \in Y^{qk} && (5.36)\\
& y_p^{qk} \in \{0,1\} && \forall k \in K, q \in Q^k, \forall p \in Y^{qk} && (5.37)\\
& x_{ij}^{\tau} \in \mathbb{N} && \forall \{i,j\} \in E, \forall \tau \in T && (5.38)
\end{aligned}$$

Constraints (5.33) enforce that for each possible network state all commodities that can be routed are routed on a single path. Edge capacity constraints (5.34)-(5.35) should be self-explanatory, and finally restriction (5.36) guarantees that the primary path is chosen for a commodity q if it is not affected in network state k.

5.4 Complexity Results and the Choice of Research Method

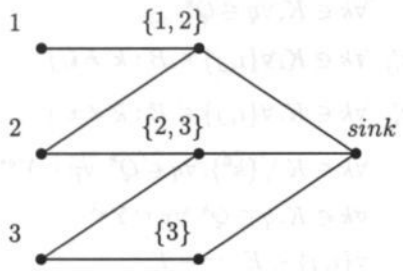
In this section we prove some complexity results, both on the non-bifurcated network loading problems and on related problems that could be part of the solution approach for these problems. We will restrict our discussion to one of the most basic network loading problems (without reliability restrictions), but the same results and ideas can be applied to all problem variants described in the above. We use the following problem definitions:

MINIMUM COVER (see Garey and Johnson [37])
INSTANCE: $S = \{s_1, s_2, \ldots, s_n\}$ and a family $\mathcal{C}$ of subsets of S, and an integer $\ell \le |C|$.
QUESTION: Does there exist a family $\mathcal{C}'$ of $\mathcal{C}$ of at most ℓ sets, such that $\cup_{c'\in\mathcal{C}'} c' = S$?

DIRECTED NON-BIFURCATED NETWORK LOADING
INSTANCE: Graph $G = (V, E)$, a set Q of commodities (s^q, t^q, d^q) with $s^q, t^q \in V$ and $d^q \in \mathbb{N}$ for each $q \in Q$, a set of capacity types T, integer capacities λ^{τ}, cost coefficients $c_{ij}^{\tau} \in \mathbb{N}$ for every $\{i,j\} \in E, \tau \in T$, and a nonnegative integer L.
QUESTION: Does there exist a feasible solution for problem $DNFM$ (5.9)-(5.13) with solution value less than or equal to L ?

Proposition 5.4.1 DIRECTED NON-BIFURCATED NETWORK LOADING *is strongly NP-complete.*

Proof. It is easy to see that the DIRECTED NON-BIFURCATED NETWORK LOADING is in NP. Next, given an instance of MINIMUM COVER, construct a graph G as follows. Let $V = S \cup \mathcal{C} \cup \{sink\}$. Let $\mathcal{C}_j$ be an element in $\mathcal{C}$. Introduce an edge $\{s_i, \mathcal{C}_j\}$ if $s_i \in \mathcal{C}_j$, and an edge $\{\mathcal{C}_j, sink\}$ for all $\mathcal{C}_j \in \mathcal{C}$ (see Figure 5.1). Next define a commodity $(s_i, sink, 1)$



$S = \{1, 2, 3\}$

$\mathcal{C} = \{\{1, 2\}, \{2, 3\}, \{3\}\}$

Figure 5.1: Transformation from MINIMUM COVER to DIRECTED NON-BIFURCATED NETWORK LOADING

for each $s_i \in S$, and define only a single capacity type with $\lambda = n$. The capacity costs are defined by $c_{\{s_i, \mathcal{C}_j\}} = |\mathcal{C}| + 1$ for all i and j and $c_{\{\mathcal{C}_j, sink\}} = 1$ for all j. Finally let $L = \ell + |S|(|\mathcal{C}| + 1)$. This is clearly a polynomial transformation.

Given an instance of MINIMUM COVER with affirmative answer, attained for subset C', define a feasible solution for DIRECTED NON-BIFURCATED NETWORK LOADING as follows. For all $s_i \in S$ there exists a $c'_i \in C'$ such that $s_i \in c'_i$. Let the commodity $(s_i, sink, 1)$ be routed along the path $s_i, c'_i, sink$ for all $s_i \in S$. Moreover, install a single unit of capacity on the edges $\{s_i, c'_i\}$ for all $s_i \in S$ and on edges $\{c'_i, sink\}$ for all $c'_i \in C'$. The total capacity costs then equal $|S|(|C| + 1) + \ell = L$.

Conversely, assume there exists a feasible solution for DIRECTED NON-BIFURCATED NETWORK LOADING with capacity costs less than or equal to L. Since each commodity must be routed away from its origin, the capacity costs are at least $|S|(|C| + 1)$. Hence, the capacity costs on the edges $\cup_j \{C_j, sink\}$ are less than or equal to ℓ. But then C' corresponding to the set of nodes c'_i for which edge $\{c'_i, sink\}$ has positive capacity yields the required subset. ■

Given this result, one can opt for several approaches to solve network loading problems. Firstly, one could try to develop approximation algorithms which guarantee that the obtained solutions are within a specified range of the optimal solutions (see Epstein [31] for some interesting approximation results for network loading problems). Secondly, one can use enumeration methods such as Branch-and-Bound or Branch-and-Cut which are

guaranteed to obtain optimal solutions. For bifurcated network loading problems this approach has proven to be quite successful (see for instance Magnanti, Mirchandani, Vachani [56], Bienstock and Günlück [16], Barahona [12]). For non-bifurcated versions, enumeration methods have also been applied (see for instance Brockmüller, Günlück and Wolsey [20] [21]) but the additional number of integer variables (due to non-bifurcation) is obviously a complicating factor. In Chapter 6 we report on a polyhedral study for the most basic versions of non-bifurcated network loading problem and the corresponding computational experience for an enumeration approach.

In this chapter we resort to a third possible approach, namely heuristic methods. In general, heuristics can be used as a robust procedure for problems of large problem size, they require less insight in problem structure, and are relatively easy to implement (cf. [1]). Moreover, the time required to obtain good solutions is usually limited and falls within the control of the designer of the heuristic algorithm. Given the diversity of the network loading problems in the current study and the limited polyhedral knowledge on the corresponding polytopes, in our opinion heuristic methods are currently the best approach to answer the research questions as mentioned in the introductory section within a reasonable amount of time. For a detailed introduction into the field of local search heuristics as well as numerous references on its applicability we refer to [1] and [63]. Here, we will confine ourselves to a brief overview of the most important concepts, using similar notation as in [1].

An instance of a network loading problem (or any combinatorial optimization problem in general) can be represented as a pair $(\mathcal{S}, f)$ where $\mathcal{S}$ denotes the set of feasible solutions for the problem and $f : \mathcal{S} \to \mathbb{R}$ is a value function that for each solution $z \in \mathcal{S}$ expresses its value $f(z)$. For network loading problems the function f represents the costs of a solution. Hence, we are interested in the globally optimal solution, that is the solution $z^* \in \mathcal{S}$ that satisfies $f(z^*) \leq f(z)$ for all $z \in \mathcal{S}$. A *neighborhood function* is now defined as a mapping $\mathcal{N} : \mathcal{S} \to 2^{\mathcal{S}}$, which for each solution $z \in \mathcal{S}$ defines a set of solutions $\mathcal{N}(z) \subseteq \mathcal{S}$ that are in some sense close to z. The set $\mathcal{N}(z)$ is called the neighborhood of solution z, and each element $y \in \mathcal{N}(z)$ is called a neighbor of z. We will assume that $z \in \mathcal{N}(z)$. The idea of (basic) local search methods is to start with an initial solution $z \in \mathcal{S}$ for the problem. Next, one tries to find a better solution for the problem by searching whether an improvement can be found among the set of neighboring solutions $\mathcal{N}(z)$. If such an improvement is found and $y \in \mathcal{N}(z)$ is the solution that yields an improvement, then one can search for a new improvement in the neighborhood $\mathcal{N}(y)$. Repeating this process until no improvement can be found yields a locally optimal solution $\hat{z} \in \mathcal{S}$, that is a solution $\hat{z} \in \mathcal{S}$ for which $f(\hat{z}) \leq f(z)$, for all $z \in \mathcal{N}(\hat{z})$. Given this description, it follows that the key challenge in the development of good local search method is to find neighborhood functions that can be searched efficiently and yield high quality solutions.

Consider the directed non-bifurcated network loading problem without reliability restrictions $DNFM$ (5.9)-(5.13), as introduced in Section 5.2. A solution for this problem is determined by both a routing scheme and a capacity assignment. Due to the cost structure of capacity installation, which we assume to be positively correlated to the required capacity on an edge, it is clear that the (minimal) costs of a solution are completely determined by a routing strategy. Stated differently, if the path from origin to destination

is fixed for each commodity and the resulting flow on arc (i,j) equals $\gamma(i,j)$, then the costs of this solution (more precisely, the costs of the cheapest solution with this routing scheme) are given by $\sum_{\{i,j\}\in E}\Gamma(i,j)(\gamma(i,j),\gamma(j,i))$, where $\Gamma(i,j)(\gamma(i,j),\gamma(j,i))$ denotes the optimal value of the integer knapsack problem $\min\{\sum_{\tau\in T} c_{ij}^{\tau}x_{ij}^{\tau} \mid \sum_{\tau\in T}\lambda^{\tau}x_{ij}^{\tau} \geq \max\{\gamma(i,j),\gamma(j,i)\}, x_{ij}^{\tau}\in\mathbb{N}, \tau\in T\}$. Although integer knapsack problems are NP-hard in general, the limited number of capacity types available in practical applications, the modularity in capacities for the different types, and relations for the cost-capacity ratio c_{ij}^{τ}/λ among the various capacity types τ give that these knapsack problems can usually be solved efficiently.

Using this idea, it follows that a solution for the directed non-bifurcated network loading problem can simply be represented by the path that is selected for each commodity. Therefore, the first idea that comes to mind is to define a neighborhood function that allows for the replacement of the selected path for a subset of the commodities. Next, we give a formal definition of this idea.

Definition 5.4.1 *Let $\mathcal{S}$ define the set of feasible solutions for the directed non-bifurcated network loading problem DNFM, where a solution itself is defined by a single path for each commodity. For all $z\in\mathcal{S}$ and all $k\in\mathbb{N}$, with $1\leq k\leq |Q|$, the k-exchange neighborhood $\mathcal{N}^k(z)$ is defined by the set of solutions $\bar{S}\subseteq\mathcal{S}$ that can be obtained from z by selecting k commodities and replacing the path for all of these k commodities by an arbitrary path in the set of feasible paths.*

Note that $\mathcal{N}^k(z)\subseteq\mathcal{N}^{k+1}(z)$ for $k=1,\ldots,|Q|-1$, for all $z\in\mathcal{S}$, and $\mathcal{N}^{|Q|}(z)$ is an exact neighborhood, that is, the optimal solution $\hat{z}\in\mathcal{N}^{|Q|}(z)$. This k-exchange neighborhood will also be referred to as *k-opt* in the sequel. Given this definition of a neighborhood, one is interested in its time complexity, that is the time required to find the best solution in the neighborhood of a solution $z\in\mathcal{S}$. First consider the k-opt neighborhood for $k=1$. Then for a solution $z\in\mathcal{S}$ we want to find the best solution among all solutions for which at least $|Q|-1$ commodities are routed on the same path as in solution z. Let $\hat{q}\in Q$, consider the routing of the commodities $q\in Q\setminus\{\hat{q}\}$ through the network fixed as in solution z, and let $\gamma(i,j)$ represent the resulting flow of these $|Q|-1$ commodities on arc $(i,j)\in A$. Then the costs of additional capacity requirement on arc (i,j) if commodity $\hat{q}$ is routed on that arc are equal to $\Gamma(i,j)(\gamma(i,j)+d^{\hat{q}},\gamma(j,i))-\Gamma(i,j)(\gamma(i,j),\gamma(j,i))$. Therefore, finding the optimal path for commodity $\hat{q}$ (i.e. the path that yields minimal costs, given the fixed routing of the remaining commodities) is a shortest path problem in which the costs on an arc in the graph are defined as the minimal costs of additional capacity if the commodity is routed on the arc. Let $f(n,m)$ denote the time complexity of a shortest path algorithm on a graph with n nodes, m arcs and nonnegative arc distances, and let $g(T,c,d)$ denote the time complexity of the integer knapsack problem Γ with capacity types set T, cost coefficients $c\in\mathbb{N}^{|T|}$, and right hand side d. Then finding the best solution in a 1-opt neighborhood of a solution requires $\mathcal{O}(|Q|\cdot f(n,m)\cdot g(T,c,\sum_{q\in Q}d^q))$ time. Shortest path problems can of course be solved in polynomial time by for instance Dijkstra's algorithm [30]. Consequently, if the calculation of the cost coefficients on the arcs of the network can be done fast in practice, this yields an efficient neighborhood structure.

Next, consider the k-exchange neighborhood for $k = 2$. Then the neighborhood $\mathcal{N}^2(z)$ of a solution $z \in \mathcal{S}$ is defined as the set of solutions for which at most two commodities may have a different path than in solution z. Let $q^1, q^2 \in Q$, $\bar{Q} = Q \setminus \{q^1, q^1\}$, and let $\gamma(i,j)$ denote the resulting flow of the fixed routing of commodities in $\bar{Q}$ on arc (i,j). Then finding the optimal routing for commodities q^1, q^2 in the network can be described by the following integer program:

$$\min \quad \sum_{\{i,j\}\in E} \sum_{\tau\in T} c_{ij}^{\tau} x_{ij}^{\tau} \tag{5.39}$$

$$\text{s.t.} \quad \sum_j f_{ij}^q - \sum_j f_{ji}^q = \begin{cases} 1 & i = s^q \\ -1 & i = t^q \\ 0 & i \neq s^q, t^q \end{cases} \qquad \forall q \in \{q^1, q^2\}, \forall i \in V \tag{5.40}$$

$$d^{q^1} f_{ij}^{q^1} + d^{q^2} f_{ij}^{q^2} + \gamma(i,j) \leq \sum_{\tau\in T} \lambda^{\tau} x_{ij}^{\tau} \qquad \forall \{i,j\} \in E \tag{5.41}$$

$$d^{q^1} f_{ji}^{q^1} + d^{q^2} f_{ji}^{q^2} + \gamma(j,i) \leq \sum_{\tau\in T} \lambda^{\tau} x_{ij}^{\tau} \qquad \forall \{i,j\} \in E \tag{5.42}$$

$$f_{ij}^{q^1}, f_{ji}^{q^1}, f_{ij}^{q^2}, f_{ji}^{q^2} \in \{0,1\},\ x_{ij}^{\tau} \in \mathbb{N} \qquad \forall \{i,j\} \in E, \forall \tau \in T \tag{5.43}$$

Unfortunately, solving this problem is NP-hard, even under the assumption that for a given flow on the arcs the costs of the solution can be determined in polynomial time. The proof of this claim employs similar techniques as Even, Itai and Shamir [32], and uses the following problem definitions.

SATISFIABILITY (see Garey and Johnson [37])
INSTANCE: A set of variables $U = \{x_1, \ldots, x_n\}$ and a set of clauses $C = \{D_1, \ldots, D_{|C|}\}$ over the variables in U.
QUESTION: Does there exist a truth assignment for the variables such that all clauses are satisfied simultaneously?

TWO COMMODITY DIRECTED NETWORK LOADING INSTANCE: A graph $G = (V, E)$, two commodities $(s^{q^1}, t^{q^1}, d^{q^1}), (s^{q^2}, t^{q^2}, d^{q^2})$ with $s^{q^i}, t^{q^i} \in V$ and $d^{q^1} \in \mathbb{N}$, $i = 1, 2$, existing flow coefficients $\gamma(i,j), \gamma(j,i) \in \mathbb{N}$ for all $\{i,j\} \in E$. A set of capacity types T, integer capacities λ^{τ}, cost coefficients $c_{ij}^{\tau} \in \mathbb{N}$ for every $\{i,j\} \in E, \tau \in T$, and a nonnegative integer K.
QUESTION: Does there exist a feasible solution for the integer program (5.39)-(5.43) with value less than or equal to K ?

Proposition 5.4.2 TWO COMMODITY DIRECTED NETWORK LOADING *is strongly NP-complete.*

Proof. It is easy to see that TWO COMMODITY DIRECTED NETWORK LOADING is in NP. Next, given an instance of SATISFIABILITY, construct an instance of TWO COMMODITY DIRECTED NETWORK LOADING as follows. For each variable x_i construct a lobe as depicted in Figure 5.2, where k_i is the number of occurrences of the literal x_i in the clauses and ℓ_i the number of occurrences of the negated literal $\neg x_i$ in the clauses (hence, w.l.o.g., $k_i + \ell_i \leq |C|$).

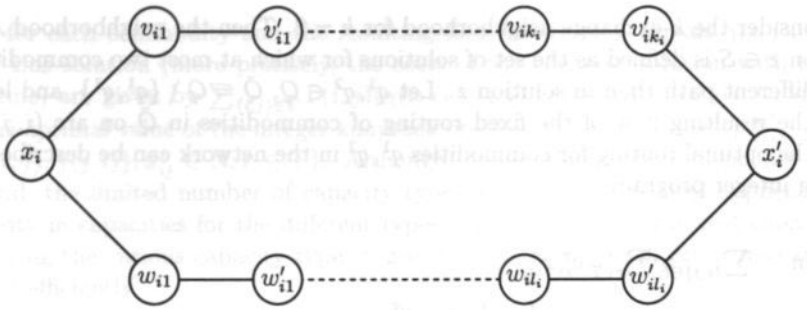


Figure 5.2: Variable Lobe

If $k_i = 0$ (or $l_i = 0$) then the upper (lower) part of the lobe is replaced by the edge $\{x_i, x'_i\}$. These lobes are connected in series, that is, there exist edges $\{s_1, x_1\}, \{x_i, x_{i+1}\}$ for $i = 1, \ldots, n-1$, and $\{x_n, t_1\}$. Next, we introduce an additional set of nodes $\{s_2, D_1, D'_1, \ldots, D_{|C|}, D'_{|C|}, t_2\}$, and edges $\{s_2, D_1\}, \{D'_i, D_{i+1}\}$ for $i = 1, \ldots, m-1$ and $\{D'_{|C|}, t_2\}$. Finally, introduce edges $\{D_j, v_{im}\}$ and $\{D'_j, v'_{im}\}$ if the assignment "x_i is true" guarantees the satisfaction of clause D_j and m refers to clause D_j. Similarly, introduce edges $\{D_j, w_{im}\}$ and $\{D'_j, w'_{im}\}$ if the assignment "x_i is false" guarantees the satisfaction of clause D_j and m refers to clause D_j. This completes the definition of the graph G, and Figure 5.3 illustrates the graph that is obtained in this way for the clauses $(x_1 \vee x_2) \wedge (x_2 \vee \neg x_3)$ on the variable set $\{x_1, x_2, x_3\}$.

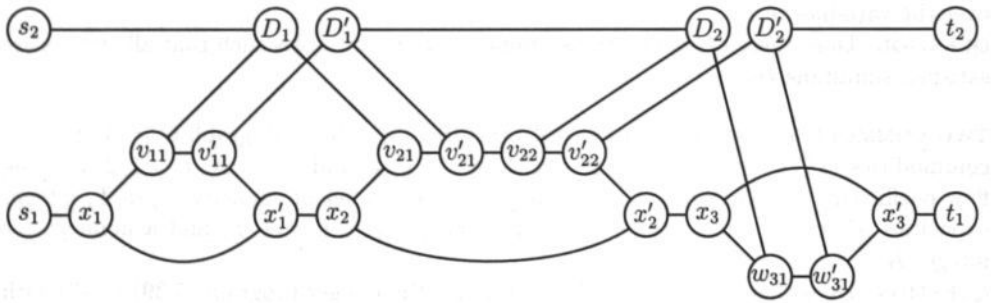


Figure 5.3: Transformation from SATISFIABILITY to TWO COMMODITY DIRECTED NETWORK LOADING

The two commodities are defined as $q^1 = (s_1, t_1, 1), q^2 = (s_2, t_2, 2)$. The existing flow coefficients on the arcs (if the corresponding edges exist) are defined as $\gamma(s_1, x_1) = 2, \gamma(x_i, v_{i1}) = \gamma(x_i, w_{i1}) = 2$, for $i = 1, \ldots, n$, $\gamma(x'_i, x_{i+1}) = 2$, for $i = 1, \ldots, n-1$, $\gamma(x'_n, t_1) = 2, \gamma(v'_{im}, v_{im+1}) = 2$, for all $i, m = 1, \ldots, k_i - 1$, $\gamma(w'_{im}, w_{im+1}) = 2$, for all $i, m = 1, \ldots, \ell_i - 1$, $\gamma(s_2, D_1) = 1, \gamma(D'_i, D_{i+1}) = 1$ for $i = 1, \ldots, n-1$, $\gamma(D'_{|C|}, t_2) = 1, \gamma(D_j, v_{im}) = 1, \gamma(D'_j, v'_{im}) = 1, \gamma(D_j, w_{im}) = 1, \gamma(D'_j, w'_{im}) = 1, \gamma(i, j) = 3$ for all reversed arcs of aforementioned arcs, and $\gamma(i, j) = 0$, otherwise.

The set of capacity types T satisfies $|T| = 1$, with $\lambda = 3$. The costs c_{ij} for capacity installation (if the edges exist) are defined as $c_{v_{im},v'_{im}} = 1$ for $i = 1,\ldots,n$ and $m = 1,\ldots,k_i$, $c_{w_{im},w'_{im}} = 1$ for $i = 1,\ldots,n$ and $m = 1,\ldots,\ell_i$. Next, $c_{v'_{ik_i},x'_i} = |C| - k_i$ if $k_i > 0$, else $c_{x_i,x'_i} = |C|$, for $i = 1,\ldots,n$. Similarly, $c_{w'_{ik_i},x'_i} = |C| - \ell_i$ if $\ell_i > 0$, else $c_{x_i,x'_i} = |C|$, for $i = 1,\ldots,n$. For all remaining edges we let $c_{ij} = n|C| + 1$. Finally, let $K = n|C|$.

If there exists a truth assignment for the variables $x_1,\ldots,x_n$ that simultaneously satisfies all clauses, then define the path for commodity q^1 as follows. If x_i is given the assignment true let $N_i = (v_{i1}, v'_{i1},\ldots,v_{ik_i}, v'_{ik_i})$; if x_i is given the assignment false, define $N_i = (w_{i1}, w'_{i1},\ldots,w_{ik_i}, w'_{ik_i})$. Then $(s_1, x_1, N_1, x'_1, x_2, N_2, x'_2,\ldots,x_n, N_n, x'_n, t_1)$ defines the path for commodity q^1. For commodity q^2, the path is defined by the sequence of nodes $(s_2, D_1, M_1, D'_1, D_2, M_2, D'_2,\ldots,D_{|C|}, M_{|C|}, D'_{|C|}, t_2)$ where M_j is defined as follows. Let $x(j)$ be the variable that occurs in clause D_j (either as literal $x(j)$ or $\neg x(j)$) and guarantees the satisfaction of clause D_j under the given truth assignment. Then $M_j = v_{im}, v'_{im}$ if m refers to clause D_j and the clause contains literal $x(j)$, and $M_j = w_{im}, w'_{im}$ if m refers to clause D_j and the clause contains literal $\neg x(j)$. The total (minimal) costs of this routing strategy equal $n|C|$ since in each lobe i the total costs are exactly $|C|$.

Conversely, assume there exists a solution for TWO COMMODITY DIRECTED NETWORK LOADING with costs less than or equal to $K = n|C|$. Then the path for commodity q^2 must be of the form $(s_2, D_1, M_1, D'_1, D_2, M_2, D'_2,\ldots,D_{|C|}, M_{|C|}, D'_{|C|}, t_2)$ with M_j either equal to v_{im}, v'_{im} or w_{im}, w'_{im} for some i and m referring to clause D_j, since otherwise capacity must be installed on one of the edges for which $c_{ij} = n|C| + 1 > K$. As a consequence, the path for commodity q^1 must be of the form $(s_1, x_1, N_1, x'_1, x_2, N_2, x'_2,\ldots,x_n, N_n, x'_n, t_1)$ with N_i equal to either $v_{i1}, v'_{i1},\ldots,v_{ik_i}, v'_{ik_i}$ or $w_{i1}, w'_{i1},\ldots,w_{ik_i}, w'_{ik_i}$ since again, otherwise capacity must be installed on an edge for which $c_{ij} = n|C| + 1 > K$.

Consider only the costs that are required to route commodity q^1, then it follows that these costs already account for the total amount K, since each lobe i requires capacity costs $|C|$. Hence, the commodity q^2 requires no additional capacity installation. But this implies that the paths for commodity q^1 and q^2 are such that M_j consists of two nodes on the path for commodity q^1. Hence, the truth assignment "x_i is true" if $N_i = v_{i1}, v'_{i1},\ldots,v_{ik_i}, v'_{ik_i}$ and "x_i is false" if $N_i = w_{i1}, w'_{i1},\ldots,w_{ik_i}, w'_{ik_i}$ yields the desired truth assignment that satisfies all clauses simultaneously. ■

Note that in a 2-opt neighborhood search one would need to solve $\mathcal{O}(|Q|^2)$ of such problems, since this represents the number of possible commodity pairs whose routing may be different from the routing in the current solution. This result therefore implies that performing a 2-opt neighborhood search may be a difficult, time-consuming task. However, since $\mathcal{N}^1(z) \subseteq \mathcal{N}^2(z)$ it follows that 2-opt may yield better solutions than 1-opt. To limit the calculation time of a 2-opt neighborhood search one could restrict the neighborhood using the following idea. Suppose we are given a (restricted) set of possible paths P^q for each commodity $q \in Q$ that can be used for the routing of the commodity. The number of different solutions that can be considered for an individual commodity pair (q^1, q^2), while keeping the routing of other commodities fixed, then equals $|P^{q^1}| \times |P^{q^2}|$. Hence, by imposing a limit on the set of paths for each commodity we can decrease the amount of time required to perform a restricted version of 2-opt. Apart from just imposing an upper

bound P on the number of paths considered for a commodity q (implying $|P^q| \leq P$) one can pose an upper bound ℓ on the number of nodes on a path in the set P^q. This leads to a restricted 2-opt neighborhood function, denoted as $\mathcal{N}^{k,P,\ell}(z)$.

Sometimes network planners are interested in symmetric routing strategies. These routing strategies imply that symmetric commodity pairs (i.e. commodities with opposite source and sink nodes) are routed on reversed paths in the graph. On the one hand, such routing strategies may be preferred for their simplicity in operation. On the other hand, although counterexamples do exist, one might expect that the additional symmetric routing restriction has little impact on the network costs if the demand data are somewhat symmetrical as well. Therefore, one can also employ a symmetric version of restricted k-opt, in which symmetric commodity pairs are routed via reversed paths.

Note that to perform a restricted (symmetric) 2-opt neighborhood search we need a method that, given a graph $G = (V, E)$ (or its directed variant $G = (V, A)$) and positive integers P and ℓ, generates a set of at most $|P|$ paths between a pair of nodes, where each path visits at most ℓ nodes. This can be done via an iterative method as follows. Let $P^n(s,t)$ denote the set of simple paths (i.e. without node repetition) from s to t with exactly n nodes on the path (including the endpoints). For initialization we let $P^2(s,t)$ be equal to the path s,t if arc $(s,t) \in A$, and $P^2(s,t) = \emptyset$, otherwise. Next, for $3 \leq n \leq \ell$ the set $P^n(s,t)$ can be obtained as follows. For all $u \in V$ such that arc $(u,t) \in A$, and all paths $p \in P^{n-1}(s,u)$ with $t \notin p$, a simple path from s to t is obtained by extending path p with the arc (u,t). This iterative process can now be used until for each commodity $q = (s^q, t^q, d^q)$ we have obtained the set $P^\ell(s^q, t^q)$ or a set $P^n(s^q, t^q)$ with cardinality $|P|$ for some $n < \ell$, whichever stopping criteria is reached first. Although this procedure may have an exponential running time in its worst case performance, in practice it yields a fast path generating method. Moreover, these path sets only have to be generated once and can then be reused in each iteration of the restricted 2-opt neighborhood search. Given the control on the amount of time needed for a neighborhood search by user defined values of P and ℓ, we prefer the usage of this restricted neighborhood function, and this is used in our current implementations, as reported in the sequel.

Finally, we mention genetic algorithms (see [41]) as a possible local search method. Genetic local search algorithms consist of a number of phases. In the *initialization* phase one constructs a set of initial solutions, referred to as the population of solutions. Next, an *improvement* phase is performed to guarantee that each individual solution in the population is locally optimal for a certain neighborhood function. In the *combination* phase one expands the population size by combining characteristics of solutions in the population to create new solutions. Consequently, an *improvement* phase is applied to guarantee that the new solutions are locally optimal as well. A *selection* phase reduces the population size to its original size according to some selection criteria. Finally, the process of combination, improvement and selection can be repeated a number of times to finally obtain high quality solutions for the combinatorial optimization problem. This method is called "genetic" since the global idea of the algorithm is to combine characteristics of solutions (so-called genetic material) to obtain new (child or offspring) solutions. The key issue that must be addressed in the development of genetic algorithms is how new offspring solutions can be created by combining the genetic material of existing (parent) solutions.

Exploiting the fact that a solution for the problem $DNFM$ or $DNPM$ is completely determined by the selected paths for the commodities (as indicated in the above), the following idea for "genetic" combination of solution arises immediately. Given two parent solutions, one could define an offspring solution as the optimal solution for the network loading problem in which the routing of each commodity is restricted to the set of paths used by either of the two parent solutions. Stated differently, the optimization problem is a directed non-bifurcated network loading problem (5.14)-(5.18) in which the cardinality of the set P^q of feasible paths for each commodity satisfies $|P^q| \leq 2$. As for the original network loading problem, this is again a strongly NP-hard problem, We use the following problem definitions to prove this claim.

MAXIMUM 2-SATISFIABILITY (see Garey and Johnson [37])
INSTANCE: Set $U = \{y_1, \ldots, y_n\}$ of variables, collection C of clauses over U such that each clause $c \in C$ satisfies $|c| = 2$, and a positive integer $R \leq |C|$.
QUESTION: Does there exist a truth assignment for the variables in U that simultaneously satisfies at least R clauses in C ?

TWO PATH DIRECTED NETWORK LOADING
INSTANCE: Graph $G = (V, E)$, a set Q of commodities (s^q, t^q, d^q), with $s^q, t^q \in V$ and $d^q \in \mathbb{N}$ for each $q \in Q$, two paths $p^q(0), p^q(1)$ from s^q to t^q in the graph G for each commodity, a set of capacity types T, integer capacities $\lambda^\tau \in \mathbb{N}$, cost coefficients $c_{ij}^\tau \in \mathbb{N}$ for every $\{i, j\} \in E, \tau \in T$, and a nonnegative integer B.
QUESTION: Does there exist a feasible solution for problem $DNPM$ (5.14)-(5.18), with path set $P^q = \{p^q(0), p^q(1)\}$ for each commodity $q \in Q$, with solution value less than or equal to B ?

Proposition 5.4.3 TWO PATH DIRECTED NETWORK LOADING *is strongly NP-complete.*

Proof. It is easy to see that TWO PATH DIRECTED NETWORK LOADING is in NP. Next, given an instance of MAXIMUM 2-SATISFIABILITY, define an instance of TWO PATH DIRECTED NETWORK LOADING as follows. Let $U^0 = U^1 = U$, and let the node set $V = \{t^1, t^2\} \cup C \cup U^0 \cup U^1 \cup U$ (where U^0 implies that there is a node for each variable with negation, U^1 implies that there is a node for each variable without negation, and U implies that there is a node for each variable in general). Moreover, define the edge set $E = \cup_{i=1}^5 E_i$, with

$$
\begin{aligned}
E_1 &= \{\{t^1, u\}, \forall u \in U^0 \cup U^1\} \\
E_2 &= \{\{c, u\}, \forall c \in C, \forall u \in U^0 : y_u \in c\} \\
E_3 &= \{\{c, u\}, \forall c \in C, \forall u \in U^1 : \neg y_u \in c\} \\
E_4 &= \{\{u, \tilde{u}\}, \forall u \in U^0 \cup U^1, \forall \tilde{u} \in U\} \\
E_5 &= \{\{u, t^2\}, \forall u \in U\}
\end{aligned}
$$

The resulting graph is depicted in Figure 5.4 for a problem instance of MAXIMUM 2-SATISFIABILITY $(x_1 \vee x_2) \wedge (\neg x_2 \vee \neg x_3)$ on the variable set $\{x_1, x_2, x_3\}$. Define the commodity set $Q = Q^1 \cup Q^2 \cup Q^3$, where $Q^1 = \{(u, t^1, n), \forall u \in U\}$, $Q^2 = \{(t^2, u, n), \forall u \in U\}$,

$Q^3 = \{(c, t^2, 1), \forall c \in C\}$. Furthermore, for $q \in Q^1$ the paths $p^q(0), p^q(1)$ are defined as $p^q(0) = (u, \bar{y}_u, t^1)$ and $p^q(1) = (u, y_u, t^1)$. For $q \in Q^2$ let $p^q(0) = p^q(1) = (t^2, u)$. Next, for $q \in Q^3$ let $a(b)$ denote the first (second) variable in clause c and let $u(a)(u(b))$ denote the way it occurs in the clause (either without or with negation). Then for $q \in Q^3$ define $p^q(0) = (c, u(a), a, t^2)$ and $p^q(1) = (c, u(b), b, t^2)$. Next, define T as a set of two capacity types, $\lambda^1 = 1, \lambda^2 = n$, $c^1_{ij} = c^1 = 2, c^2_{ij} = c^2 = 2n - 1$, for all $\{i, j\} \in E$. Finally, let $B = 3|U|c^2 + nc^1 + (n - R)c^1$. This is clearly a polynomial transformation. It remains to show that an instance for MAXIMUM 2-SATISFIABILITY with affirmative answer yields an instance for TWO PATH DIRECTED NETWORK LOADING with affirmative answer, and vice versa.

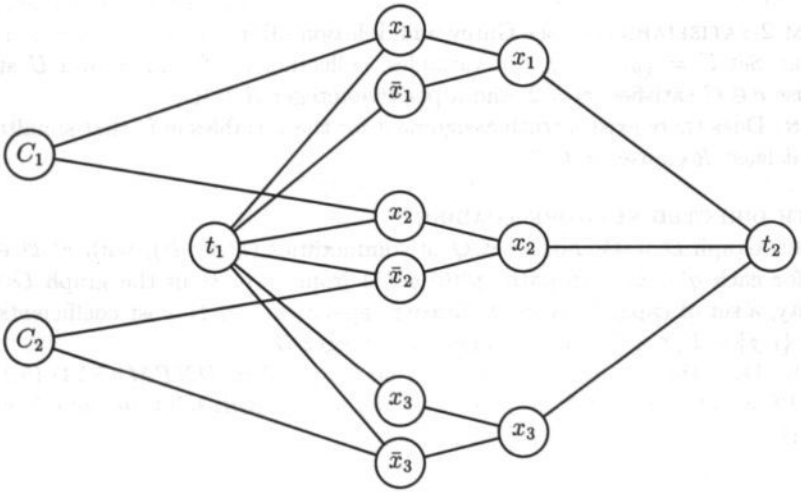


Figure 5.4: Transformation from MAXIMUM 2-SATISFIABILITY to TWO PATH DIRECTED NETWORK LOADING

Consider an instance for MAXIMUM 2-SATISFIABILITY with affirmative answer, that is there exists a truth assignment for the variables in U that satisfies at least R clauses simultaneously. Then define a solution for TWO PATH DIRECTED NETWORK LOADING as follows. For $q \in Q^1$, if the variable y_u is assigned the value true ($y_u = 1$) then choose path $p^q(1)$, else the variable is assigned the value false ($y_u = 0$) and choose path $p^q(0)$. For $q \in Q^2$, there is no real choice, since both paths $p^q(0), p^q(1)$ are equivalent. Finally, for $q \in Q^3$, if clause c is satisfied by the first literal $u(a)$, then choose path $p^q(0)$, else choose path $p^q(1)$. Given this choice of routing, let $\bar{E}_i \subseteq E_i$ denote the set of edges in E_i with positive flow. Then we define the capacity variables on the edges as follows. For $\{i, j\} \in \bar{E}_2 \cup \bar{E}_3$ let $x^1_{ij} = 1, x^2_{ij} = 0$. For $\{i, j\} \in \bar{E}_1 \cup \bar{E}_5$ let $x^1_{ij} = 0, x^2_{ij} = 1$. Finally, for $\{i, j\} \in \bar{E}_4$, if $\{i, j\}$ is on a path for a commodity $q \in Q^1$, then let $x^1_{ij} = 0, x^2_{ij} = 1$, else let $x^1_{ij} = 1, x^2_{ij} = 0$. Then it follows easily that the total costs of this solution are less than or equal to B.

Conversely, if there exists a solution (routing and capacity assignment) for TWO PATH DIRECTED NETWORK LOADING with costs less than or equal to B, then, independent of the routing choices for $q \in Q^1 \cup Q^2$, the minimal costs on edges in $E_1 \cup E_4 \cup E_5$

required to make this routing feasible equals $3|U|c^2$. Similarly, independent of the choice of routing for $q \in Q^3$ the capacity requirement on edges $\{i,j\} \in E_2 \cup E_3$ equals nc^1. In conclusion, at most $(n-R)c^1$ is spent on capacity assignment on the edges. Since, $c^2 = 2n - 1 > c^1(n-R)$ (w.l.o.g. assume that $R > 1$) it follows that at most $n - R$ units capacity of type 1 are installed on edges $\{i,j\} \in E_4$ in the solution. But this implies that at most $n - R$ commodities from the set Q^3 are routed on edges which are not used for the routing of commodities in Q^1. Hence, at least R commodities from Q^3 are routed on edges $\{i,j\} \in E_4 \cup E_5$ which are also used for the routing of commodities in Q^1. As a result, the truth assignment such that $y_u = 1$ if $p^q(1)$ is chosen for $q \in Q^1$, and $y_u = 0$ if $p^q(0)$ is chosen for $q \in Q^1$ yields the desired truth assignment. ■

The fact that this is an NP-hard problem does not immediately imply that a genetic algorithm that uses optimization in its combination phase is computationally infeasible (see Kolen [52] for a counterexample). Still, we have chosen to perform our research with the implementation of the restricted k-opt neighborhood structures.

5.5 UMBRIA: A Decision Support System for Network Loading Problems

In this section we introduce UMBRIA: a decision support system for network loading problems. This software tool is developed to guide network planners in the network design phase. Currently, it supports the analysis of network loading problems as described in Sections 5.2 and 5.3, and employs local search algorithms based on the restricted k-opt neighborhood functions.

From a developer's point of view, the most important requirement is the fact that the software must be easily accessible for network planners, since otherwise it will not be used. We have tried to satisfy this restriction by the development of a user-friendly interface that uses standard windows functionalities, thereby increasing its self-explanatory value and restricting its learning curve. From a user's point of view, the functionalities of UMBRIA can be partitioned into three categories, namely into input, algorithmic, and output functionalities, each of which form the basis for one of the subsequent subsections.

5.5.1 Input Functionalities of UMBRIA

Due to the variety in network loading problems that can be analyzed with UMBRIA a lot of input data are required. In total, seven different input categories are distinguished, and, depending on the exact problem type at hand, all data for the corresponding subset of categories must be made available. UMBRIA contains a dedicated input sheet for each individual category which allows the user to enter the required data. The first input category consists of the problem characteristics that define the exact problem type that should be analyzed, and Figure 5.5 shows the associated input sheet.

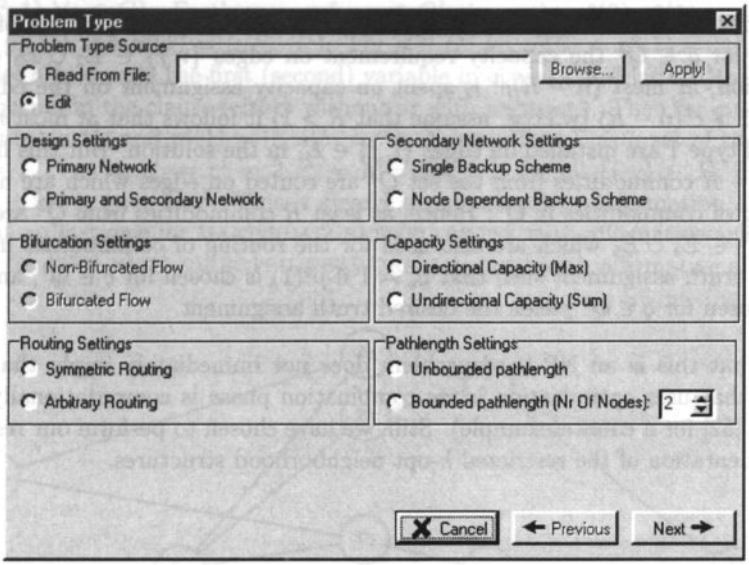


Figure 5.5: Problem Type Input Page

As can be seen from the figure, the user can select whether a primary network or both primary and secondary network should be constructed, in which case the user can also select whether a single backup scheme (single secondary path, as in Subsection 5.3.1) or a node dependent backup scheme (multiple secondary paths, as in Subsection 5.3.2) should be adapted. The capacity settings allow the user to specify whether the network loading problem under analysis is directed or undirected. Finally some additional restrictions on solutions for the network loading problem can be imposed. The routing settings indicate whether any commodity can be routed arbitrarily through the graph, or instead, symmetric commodity pairs (that is, commodity from s to t and the commodity from t to s) should be routed on reversed paths through the network. The path length settings allow the user to restrict the number of nodes on the selected path for a commodity.

The second input category describes the graph on which the problem is defined, and Figure 5.6 depicts the corresponding input sheet on which the user can specify the nodes and edges. Similar input sheets are available for the remaining input categories, which include the demand data, the different capacity types, cost data, possible routing restrictions on primary or secondary paths, and a specification of the set of failure states. All of these input data can be stored in and retrieved from files, such that data can be reused for future purpose. To simplify reuseability, input categories of distinct problem instances can be combined to obtain new problem definitions, provided that these partial input data match. UMBRIA contains a module to perform a data consistency verification.

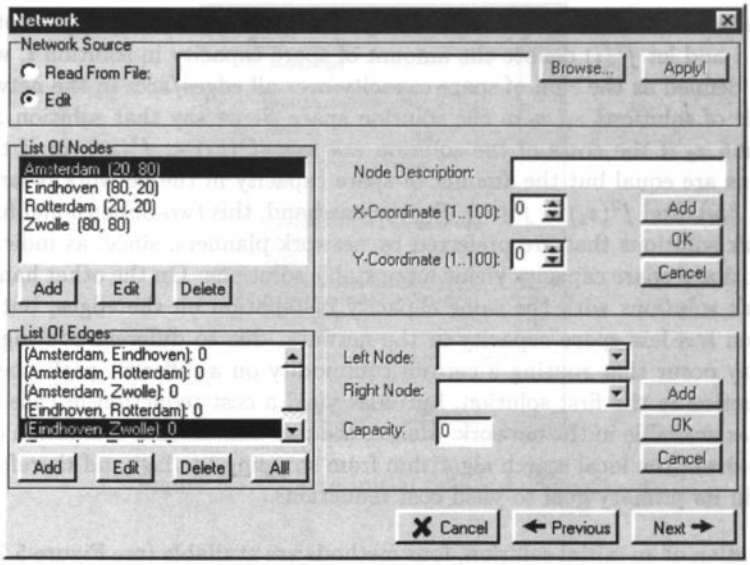


Figure 5.6: Network Input Page

5.5.2 Algorithmic Functionalities of UMBRIA

Currently, UMBRIA uses an iterative improvement local search. First an initial solution is created. Next, we perform a neighborhood search based on the restricted k-opt neighborhood function described in Section 5.4 to find the best solution in the neighborhood. If an improvement (i.e. a better solution) is found, we search in the neighborhood of this new solution. This process is repeated until a locally optimal solution is found, i.e. no improvements can be found in the neighborhood of the solution. However, we have also provided a so-called kick operation. Instead of starting a new search with a completely new solution from scratch, this kick operation replaces the selected paths for a user-defined number of commodities by random paths. Thereby it allows for a small disturbance of a locally optimal solution, which may then function as input for iterative improvement.

The definition of improvement as used in the local search method leaves room for discussion. The most simple variant of improvement is based on the idea that for any pair of solutions z_1, z_2 in the solution space $\mathcal{S}$, it holds that solution z_1 is better than solution z_2 if the costs of solution z_1 are less than the costs of solution z_2. However, apart from the cost criteria one could also judge a network solution on the amount of spare capacity which is available in the network. This spare capacity is a measure for the amount of extra telecommunication traffic flow that the network can handle without additional capacity installation, hence, it can be seen as a measure for the network stability under future demand growth. Therefore, instead of using a one-dimensional function for the evaluation and comparison of solutions we propose a two-dimensional function, that incorporates both the costs of capacity installation of a solution and the amount of spare

capacity in the network for a solution. More precisely, for $i \in \mathcal{S}$, let $f^1(i)$ denote the costs of solution i and let $f^2(i)$ denote the amount of spare capacity in solution i, where spare capacity is defined as the sum of spare capacity over all edges/arcs in the network. Then for any pair of solutions z_1, z_2 in the solution space $\mathcal{S}$, we say that solution z_1 is better than solution z_2 if the costs of the solution are less, $f^1(z_1) < f^1(z_2)$, or if the costs of the solutions are equal but the amount of spare capacity in the network is larger, hence, $f^1(z_1) = f^1(z_2)$, and $f^2(z_1) > f^2(z_2)$. On the one hand, this two-dimensional function will find network solutions that are preferred by network planners, since, as indicated in the above, additional spare capacity yields more stable solutions. On the other hand, consider two network solutions with the same capacity installation on the edges, but where the first solution has less spare capacity in the network, due to different routing strategies. Then it may occur that routing a certain commodity on a different path does not yield a cost reduction in the first solution, but does yield a cost reduction in case more spare capacity was available in the network. Hence, the two-dimensional evaluation of solutions may also prevent the local search algorithm from stopping too fast and therefore help the algorithm in its primary goal to yield cost reductions.

For the creation of an initial solution, four methods are available (see Figure 5.7 for a visualization of the options as incorporated in UMBRIA). The shortest path method selects a shortest path in the graph for each commodity, where the length of a path is defined as the number of arcs (or nodes) on the path. This method is based on the following idea. Consider a commodity $q \in Q$ and a path $p \in P^q$. Then the commodity uses a certain amount of capacity on each edge/arc of the path. Let the capacity consumption of the commodity for the selected path therefore be defined as the demand of the commodity times the number of arcs on path p. Similarly, we can define the total capacity consumption in the network by accumulating the individual commodity consumption values. Although no perfect relation holds in general, one does expect that network solutions with lower capacity consumption require less capacity installation, and are therefore cheaper. The shortest path method is based on this idea since a shortest path for a commodity yields the lowest possible capacity consumption for a commodity.

The second method as indicated in Figure 5.7 is a random method that selects an arbitrary path for each commodity. This may be particularly helpful if the local search algorithm is performed multiple times to have different starting solutions for the search process. The third method and fourth method are more constructive in the following sense. Suppose that a subset of commodities is routed in the network then the resulting flow on the edges/arcs already require a certain capacity installation. Because capacity can only be installed in certain discrete amounts this implies that there may exist some spare capacity for this partial solution. A commodity that is not yet routed could take advantage of this spare capacity. Hence, given the routing of a subset of the commodities one wants to find the best routing for the next commodity in the order. This one commodity network loading problem is a shortest path problem, where the costs on an arc are defined as the costs of additionally routing the commodity on the arc. The third and fourth method therefore add the commodities one by one, each time finding the cheapest routing for the new commodity given the partial solution. The third method (LOCI: Largest commodity Order Cheapest Insert) orders the commodity according to non-increasing demand, whereas the fourth method (ROCI: Random commodity Order Cheapest Insert)

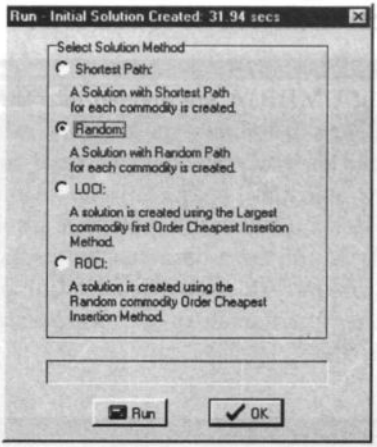


Figure 5.7: Create Initial Solution Window

uses a random order on the commodities.

Figure 5.8 visualizes the neighborhood functions that are supported by UMBRIA. Given user-defined values for the parameters P (maximum number of generated paths per commodity) and ℓ (maximum number of nodes on generated paths) the k-opt ($k = 1, 2$) selection perform an iterative improvement search based on the restricted neighborhood function $\mathcal{N}^{k,P,\ell}(z)$. Note that 1-opt and 2-opt are not available to the user if the routing of commodities is required to be symmetric. Instead, the symmetric versions of these neighborhood functions are always available.

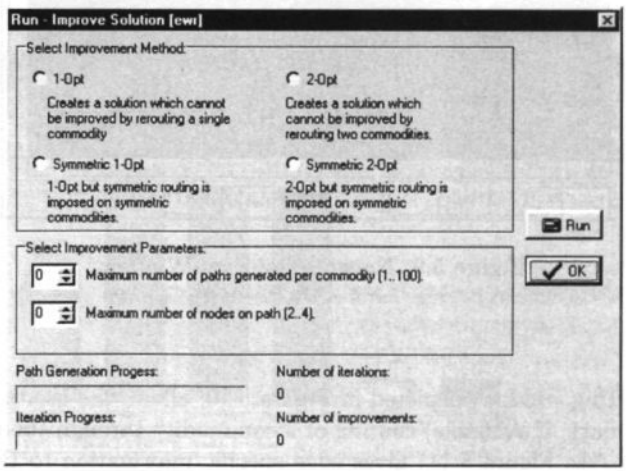


Figure 5.8: Improve Solution Window

5.5.3 Output Functionalities of UMBRIA

The output functionalities of UMBRIA are divided into three different visualizations of the generated solutions. Figure 5.9 shows the network solution window, which allows the user to view global solution statistics such as total network costs, total capacity installation, total traffic flow and total spare capacity. Moreover, a visualization of the network allows the user to view the network topology of a solution and inspect such data per individual edge as well. All of these data can be displayed or hidden, according to the user's specific needs. Moreover, since the visualization of large, dense networks may become troublesome, zooming functionalities are incorporated, as well as the possibility to display any subset of the edges.

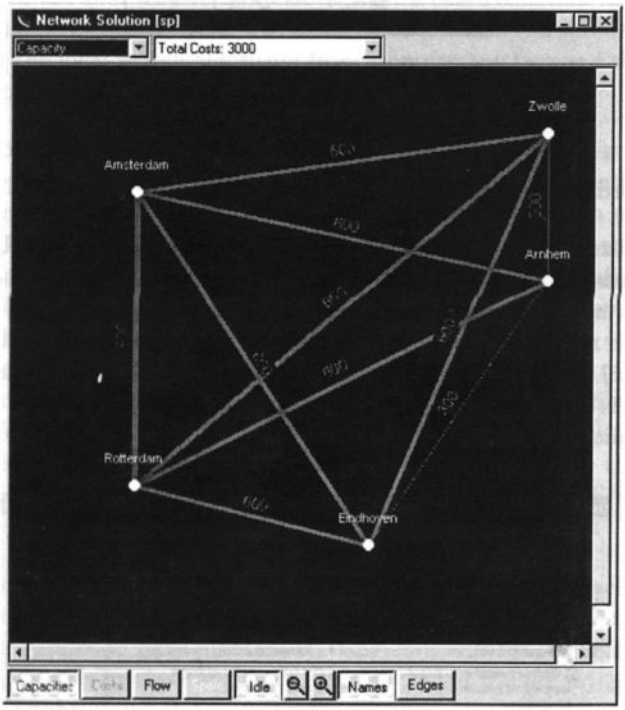


Figure 5.9: Network Solution Window

The demand routing window, depicted in Figure 5.10, allows the user to inspect the primary (and secondary, if available) routing of a commodity through the network. Finally, the link window (see Figure 5.11) gives edge specific information to the user. This includes data such as the costs and capacity on the edge, as well as a specification of these numbers per capacity type, and the flow and spare capacity on the edge. Next, one can monitor the different commodities that visit the edge in both directions. These solution statistics may help network planners in the comparison of different network solutions.

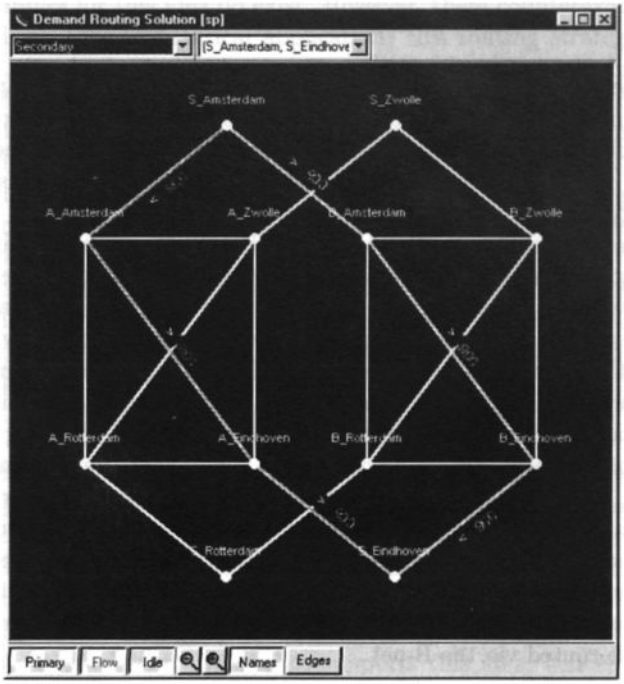


Figure 5.10: Demand Routing Window

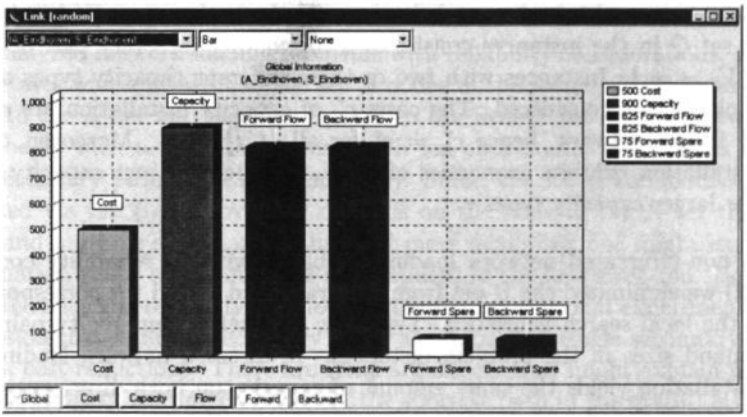


Figure 5.11: Link Window

5.6 Conclusions

The research questions as posed in the introduction were part of a planning project for ATM-backbone networks at KPN Research, Leidschendam, and they provided us with a number of real-life problem instances. First we explain the structure of these instances (see also Figure 5.10 for an example graph). For historical reasons, the existing hierarchical network topology contains two identical, separate backbone networks. These networks are cliques and are referred to as the A-net and B-net. For each node in the A-net that serves some geographical area there exists a "brother" node in the B-net that serves the same geographical area . Moreover, the lower layers of the network hierarchy in a certain geographical area are both connected to the corresponding backbone switch in the A-net and B-net, and these lower layers are represented by a single node in the example figure (S-nodes). Historically, if a traffic demand generated in a lower layer of the network should be routed from an origin area to a destination area then the following network routing strategy was typically used. In case no network component breakdown occurred in the A-net backbone, the traffic demands was routed directly from the lower layer (switching network) to a backbone switch of the A-net that corresponds to the origin area. Next, it was routed via the A-net to a backbone node of the destination area, after which it was routed to the switching network in the destination area. In case this routing was no longer feasible due to a network component failure in the A-net the corresponding traffic demand would be routed via the B-net.

The network design issues discussed in this chapter are based on this underlying network topology. Hence, the node set of a graph $G = (V, E)$ corresponding to a problem instance can be partitioned into three parts $V = V_S \cup V_A \cup V_B$, with $|V_S| = |V_A| = |V_B|$. The set V_S denotes the set of demand nodes that are source or sink of a commodity. Each of these nodes is connected to exactly one node in V_A and one node in V_B. Moreover, the nodes in V_A are connected via a clique graph, and the same holds for the nodes in V_B. The instances we obtained were defined on graphs in the range $|V_S| = 4, \ldots, 8$. The commodity set Q in the instances consist of a commodity for every pair of nodes (s, t) with $s, t \in V_S$, $s \neq t$. Instances with two or three different capacity types arising from ATM-technology were considered. The costs c^τ_{ij} of capacity installation are equal for all edges $\{i, j\}$ in the network, hence $c^\tau_{ij} = c^\tau$ for all $\{i, j\} \in E$. Moreover, the costs of capacity installation exhibits economies of scale, i.e. the costs per capacity ratio c^τ/λ^τ decreases for larger capacity types τ.

To analyze non-bifurcated network loading problems without reliability considerations (Section 5.2) we eliminated the B-net from the graph and solved the corresponding problems using the local search heuristics. Typically, symmetric commodity pairs had comparable demand sizes in the problem instances. In directed network loading problems capacity installation yields the same amount of capacity on both directions on an edge. This might lead to the idea that symmetric routing strategies are sensible since if a commodity is routed on an arc it requires a certain capacity installation on the edge, hence that same capacity is available in the other direction on the edge and this might as well be used by the reversed commodity with comparable demand size. From a theoretical point of view, even if symmetric commodity pairs have exactly the same demand size, the

optimal routing strategy does not necessarily have to be a symmetric routing strategy, and counterexamples for this idea do exist. However, these counterexamples may well be somewhat artificial, thus in practical applications this routing strategy may be a good choice. In fact, in all the instances considered in our study with directional capacity we could find no reduction in costs by allowing arbitrary routing strategies instead of symmetric routing strategies. In case of an undirected capacity problem the argument in favor of symmetric routing is less apparent, and in fact, the example problem instance discussed in Chapter 1 is an example for which arbitrary routing strategies may yield cost reductions as compared to symmetric cost reductions. However, in the real-life problem instances cost reductions could hardly ever be obtained by allowing arbitrary as opposed to symmetric routing strategies. Moreover, in the rare cases where such a cost reduction was possible, the resulting routing strategies were not very stable under growing demand sizes of the commodities. Therefore, we conclude that symmetric routing yields good network topologies in practice in case the demand sizes of symmetric commodity pairs are comparable.

Another research topic involves the influence of an upper bound on the number of nodes on the selected path from the source of a commodity to the sink of a commodity. Without the incorporation of such an upper bound, the paths used in the solutions found by our heuristics usually already contained a small number of nodes. This can well be explained by the fact that short paths lead to less capacity consumption of a commodity in a network and this tends to lead to a lower capacity requirement. In fact, the definition of improvement in our local search heuristics supports this idea, hence, for different solutions with equivalent capacity costs the solution that uses shorter paths for commodities is preferred. Therefore, adding an upper bound on the path length used in a routing strategy was expected to have little impact on the network costs, and this conclusion is supported by our computations. Since one or two additional intermediate nodes on a path as compared to the shortest possible path is usually acceptable, this leads to the conclusion that realistic upper bounds on the path length in routing strategies cause no significant higher network costs.

Next we analyzed network loading problems with reliability considerations, where the set of failure states was equivalent to the set of nodes in the A-net. The primary paths were restricted to the A-net and secondary paths to the B-net. We compared two situations, namely models with a single secondary path for each commodity versus models with multiple secondary paths for each commodity. Since, the set of commodities that needs to be routed via the B-net obviously depends on the node in the A-net that is subject to failure and multiple secondary paths yield more flexibility, one might expect that the second situation yields cost reductions as compared to the first situation. However, again such cost reduction were hardly ever found in our computational experiments, leading to the conclusion that multiple secondary paths as opposed to single secondary paths yields no network cost reductions. There are some arguments that might explain this empirical result. Firstly, direct routing (routing via the shortest path in the graph) is often applied for the primary paths. Hence, these paths often contain a small number of nodes from the A-net. This implies that the extra flexibility that can be obtained in the second model is limited since secondary paths are only required for each potential failure node on the primary path. Secondly, the capacity required in the B-net for the secondary path for a

certain failure state might as well be used for a secondary path of the same commodity for a different failure state.

Chapter 6

Polyhedral Results for the Edge Capacity Polytope

6.1 Introduction

This chapter, based on Van Hoesel, Koster, van de Leensel and Savelsbergh [77], considers the edge capacity polytope. This polytope arises as an important substructure in all of the network loading models described in Chapter 5. These models contain capacity constraints on the edges of the network, which guarantee that the amount of capacity installed on an edge of the network suffices to accommodate the amount of flow on the edge for a possible network state, i.e. failure or non-failure state. Although the different models contain somewhat different edge capacity constraints, from a mathematical point of view these restrictions are the same. In all cases one is given a set of capacity types, whose cumulative integer capacity must be greater than or equal to the cumulative demand of the commodities that are routed on the edge. In this chapter we focus on this general model, and study the polyhedral structure of the associated edge capacity polytope. We restrict ourselves to a single capacity type, although many of the ideas presented in this chapter can be extended in case multiple capacity types are available.

The edge capacity polytope with a single capacity type can be viewed as a 0–1 knapsack problem with a single integer variable representing the capacity of the knapsack. The closely related knapsack problem with a single *continuous* capacity variable is studied by Marchand and Wolsey [58]. They employ valid inequalities of the standard 0–1 knapsack problem (see Balas [10], Hammer, Johnson and Peled [45], Wolsey [79]) to obtain valid inequalities for the extended model by projection and lifting. The edge capacity polytope itself has also been studied by Brockmüller, Günlück and Wolsey [20],[21], who derive valid and facet defining inequalities for the edge capacity polytope. Magnanti, Mirchandani and Vachani [55] study the version of the polytope in which the binary variables are relaxed to real variables and derive a complete description of the corresponding polytope.

In this chapter we derive various new results for the edge capacity polytope. In Section 6.2 we recall the non-bifurcated network loading problems as stated in Chapter 5, and formu-

late the associated edge capacity models corresponding to individual capacity constraints. Although valid inequalities for individual constraint models obviously yield valid inequalities for the overall model, in Section 6.3 we derive a much stronger result, which implies that facet defining inequalities for the edge capacity polytope (often) correspond to facet defining inequalities for the network loading problem. In Section 6.4 we derive some basic results on facet defining inequalities for the edge capacity polytope, which facilitate the subsequent analysis. Section 6.5 introduces lower convex envelop inequalities as a general framework for valid inequalities for the edge capacity polytope.

Given the interpretation of the edge capacity polytope as a knapsack polytope with variable capacity, Section 6.6 shows how any valid inequality for related knapsack polytopes can be transformed into a valid inequality for the edge capacity polytope by integer lifting. Integer lifting is usually a complicated process, but in Section 6.7 we show that for cover inequalities, this lifting process can be done efficiently, and moreover, that the resulting inequalities include the so-called c-strong inequalities developed by Brockmüller, Günlück and Wolsey [20]. Some new properties that indicate the importance of c-strong inequalities in the description of the edge capacity polytope are also mentioned. In Section 6.8 we discuss the directed edge capacity polytope, that is the edge capacity polytope with a capacity constraint for both arcs corresponding to a single edge in the network. New valid inequalities and lifting results are obtained. Finally, the computational importance of the developed theory is the subject of Section 6.9.

6.2 Models for Network Loading Problems

Let $G = (V, E)$ be an undirected connected graph with node set V and edge set E. Given the graph G we define the arc set A, which contains two directed arcs (i, j) and (j, i) for all edges $e = \{i, j\} \in E$. Let Q be a set of demands (commodities). Each element $q \in Q$ is a triple (s^q, t^q, d^q), with $s^q, t^q \in V$, $s^q \neq t^q$, representing a commodity with demand size $d^q \in \mathbb{N}$ that must be routed from source node s^q to sink node t^q on a single path through the network. To route a set of commodities on an arc, sufficient capacity must be available on the corresponding edge. The capacity on an edge is determined by the number of capacity units installed on the edge, where each unit has a base capacity $\lambda \in \mathbb{N}$. The goal is to minimize the costs of the installed capacity in the network while ensuring that all commodities can be routed from source to sink simultaneously.

We assume that for each commodity $q \in Q$ there exist at least two node-disjoint paths from source node to sink node (node-disjoint, except for the nodes s^q and t^q). If this assumption is not satisfied, the graph G contains a separating vertex, hence the problem can be decomposed into smaller problems that do satisfy the assumption.

To formulate this problem as an integer program, let $x_{ij} \in \mathbb{N}$ be the number of capacity units installed on edge $\{i, j\}$, and let f_{ij}^q be a binary variable indicating whether the commodity $q \in Q$ is routed via arc $(i, j) \in A$ or not. If c_{ij} represents the costs per base

capacity unit on edge $\{i,j\} \in E$, then the model reads:

$$\min \quad \sum_{\{i,j\}\in E} c_{ij} x_{ij} \tag{6.1}$$

$$\text{s.t.} \quad \sum_j f_{ij}^q - \sum_j f_{ji}^q = \begin{cases} 1 & \text{if } i = s^q \\ -1 & \text{if } i = t^q \\ 0 & \text{otherwise} \end{cases} \quad \forall q \in Q, \forall i \in V \tag{6.2}$$

$$\lambda x_{ij} \geq \sum_{q\in Q} d^q (f_{ij}^q + f_{ji}^q) \qquad \forall \{i,j\} \in E \tag{6.3}$$

$$f_{ij}^q, f_{ji}^q \in \{0,1\},\ x_{ij} \in \mathbb{N} \qquad \forall q \in Q,\ \forall \{i,j\} \in E \tag{6.4}$$

This model is called the undirected non-bifurcated flow model, and the corresponding set of feasible solutions is denoted $UNFM$. The capacity on an edge is *undirected* because installed capacity can be used by traffic in both directions, i.e. the required capacity on an edge is determined by the sum of forward and backward flow on the edge. It is called *non-bifurcated* since the demand of a commodity has to be routed on a single path (i.e. the demand cannot be bifurcated). Finally, *flow* variables on individual arcs are used to model the routing of a commodity from source node to sink node.

Instead of using flow variables on individual edges to model routing restrictions, one can also use binary variables z_p^q representing whether a certain path $p \in P^q$ (the set of all possible paths for the commodity q) is used to route the commodity q from source node s^q to sink node t^q. We assume that P^q only contains simple paths, that is paths that visit each node at most once. If $P_{ij}^q \subseteq P^q$ denotes the set of paths for commodity q that contain arc (i,j), then this leads to the following undirected non-bifurcated path model with feasible solution set $UNPM$.

$$\min \quad \sum_{\{i,j\}\in E} c_{ij} x_{ij} \tag{6.5}$$

$$\text{s.t.} \quad \sum_{p\in P^q} z_p^q = 1 \qquad \forall q \in Q \tag{6.6}$$

$$\lambda x_{ij} \geq \sum_{q\in Q} \sum_{p\in P_{ij}^q \cup P_{ji}^q} d^q z_p^q \qquad \forall \{i,j\} \in E \tag{6.7}$$

$$z_p^q \in \{0,1\},\ x_{ij} \in \mathbb{N} \qquad \forall q \in Q,\ \forall p \in P^q,\ \forall \{i,j\} \in E \tag{6.8}$$

Depending on the exact application and level of aggregation, capacity that is installed on edges in the network can also be *directed*, i.e. each unit of capacity installed on an edge $\{i,j\}$ gives a capacity of λ on both corresponding arcs (i,j) and (j,i), and capacity consumption is directed as well. This leads to the following directed non-bifurcated flow model, with feasible solution set $DNFM$.

$$\min \quad \sum_{\{i,j\}\in E} c_{ij} x_{ij} \tag{6.9}$$

$$\text{s.t.} \quad \sum_j f_{ij}^q - \sum_j f_{ji}^q = \begin{cases} 1 & \text{if } i = s^q \\ -1 & \text{if } i = t^q \\ 0 & \text{otherwise} \end{cases} \quad \forall q \in Q, \forall i \in V \tag{6.10}$$

$$\lambda x_{ij} \geq \sum_{q\in Q} d^q f_{ij}^q \qquad \forall \{i,j\} \in E \tag{6.11}$$

$$\lambda x_{ij} \geq \sum_{q \in Q} d^q f^q_{ji} \qquad \forall \{i,j\} \in E \tag{6.12}$$

$$f^q_{ij}, f^q_{ji} \in \{0,1\},\ x_{ij} \in \mathbb{N} \qquad \forall q \in Q,\ \forall \{i,j\} \in E \tag{6.13}$$

Similar to the undirected case, one can model the directed case using path variables. This directed non-bifurcated path model, with feasible solution set $DNPM$ reads:

$$\text{min} \quad \sum_{\{i,j\} \in E} c_{ij} x_{ij} \tag{6.14}$$

$$\text{s.t.} \quad \sum_{p \in P^q} z^q_p = 1 \qquad \forall q \in Q \tag{6.15}$$

$$\lambda x_{ij} \geq \sum_{q \in Q} \sum_{p \in P^q_{ij}} d^q z^q_p \qquad \forall \{i,j\} \in E \tag{6.16}$$

$$\lambda x_{ij} \geq \sum_{q \in Q} \sum_{p \in P^q_{ji}} d^q z^q_p \qquad \forall \{i,j\} \in E \tag{6.17}$$

$$z^q_p \in \{0,1\},\ x_{ij} \in \mathbb{N} \qquad \forall q \in Q,\ \forall p \in P^q,\ \forall \{i,j\} \in E \tag{6.18}$$

Both the directed and the undirected version of the problem are strongly NP-hard (see Chapter 5). In this chapter we focus on the polyhedral structure of the associated polytopes. More precisely, we study the convex hull of sets related to individual edge capacity constraints in the formulations. Hence, we define the following sets, which are defined by a single (denoted with X) or double (denoted with Y) edge capacity constraint.

$$X^{UF}_{ij} = \{(x,f) \in \mathbb{N} \times \{0,1\}^{2|Q|} : \lambda x_{ij} \geq \sum_{q \in Q} d^q (f^q_{ij} + f^q_{ji})\}$$

$$X^{DF}_{ij} = \{(x,f) \in \mathbb{N} \times \{0,1\}^{|Q|} : \lambda x_{ij} \geq \sum_{q \in Q} d^q f^q_{ij}\}$$

$$Y^{DF}_{ij} = \{(x,f) \in \mathbb{N} \times \{0,1\}^{2|Q|} : \lambda x_{ij} \geq \sum_{q \in Q} d^q f^q_{ij},\ \lambda x_{ij} \geq \sum_{q \in Q} d^q f^q_{ji}\}$$

$$X^{UP}_{ij} = \{(x,z) \in \mathbb{N} \times \{0,1\}^{\sum_{q \in Q} |P^q_{ij}| + |P^q_{ji}|} : \lambda x_{ij} \geq \sum_{q \in Q} \sum_{p \in P^q_{ij} \cup P^q_{ji}} d^q z^q_p,\ \sum_{p \in P^q_{ij} \cup P^q_{ji}} z^q_p \leq 1, \forall q \in Q\}$$

$$X^{DP}_{ij} = \{(x,z) \in \mathbb{N} \times \{0,1\}^{\sum_{q \in Q} |P^q_{ij}|} : \lambda x_{ij} \geq \sum_{q \in Q} \sum_{p \in P^q_{ij}} d^q z^q_p,\ \sum_{p \in P^q_{ij}} z^q_p \leq 1, \forall q \in Q\}$$

$$Y^{DP}_{ij} = \{(x,z) \in \mathbb{N} \times \{0,1\}^{\sum_{q \in Q} |P^q_{ij}| + |P^q_{ji}|} : \lambda x_{ij} \geq \sum_{q \in Q} \sum_{p \in P^q_{ij}} d^q z^q_p,\ \lambda x_{ij} \geq \sum_{q \in Q} \sum_{p \in P^q_{ji}} d^q z^q_p,\ \sum_{p \in P^q_{ij} \cup P^q_{ji}} z^q_p \leq 1\}$$

Obviously any valid inequality for these polytopes is valid for the corresponding original problem. In Section 6.3 we prove the stronger result that for the undirected models any non-trivial facet defining inequality for these polytopes is also a facet defining inequality for the corresponding original problem. For the directed models the same result holds for the edge models that incorporate capacity constraints in both directions on the edge.

6.3 The Strength of Facets of the Edge Capacity Polytope for Network Loading Problems

In this section we state some basic results about the dimension of polytopes related to the problems introduced in the previous section. More importantly, we show that non-trivial facet defining inequalities for the polytopes related to a single edge constraint are non-trivial facet defining inequalities for the polytopes corresponding to the original problem.

6.3.1 Path Formulation

Proposition 6.3.1 *The dimension of both conv(UNPM) and conv(DNPM) are equal to* $|E| + \sum_{q \in Q}(|P^q| - 1)$.

Proof. The number of edge capacity variables equals $|E|$ and the number of path variables equals $\sum_{q \in Q} |P^q|$. Since the number of linearly independent equality constraints equals $|Q|$, this leads to an upper bound on the dimension of $|E| + \sum_{q \in Q}(|P^q| - 1)$. Next, we state $1 + |E| + \sum_{q \in Q}(|P^q| - 1)$ affinely independent feasible solutions, which proves our claim. In the first solution each commodity $q \in Q$ is routed via an arbitrarily chosen path $\hat{p} \in P^q$, and the capacity equals the total flow on an edge rounded up to the nearest multiple of λ. Given this solution we can install an extra capacity unit on each edge, which yields another $|E|$ affinely independent solutions. Finally, for each commodity $q \in Q$ and each path $p \in P^q \setminus \{\hat{p}\}$ we construct a solution by keeping the routing of all other commodities fixed as in the first solution, but replacing path $\hat{p}$ by path p for commodity q, and installing additional capacity if needed. The $\sum_{q \in Q}(|P^q| - 1)$ vectors that are obtained are affinely independent since each solution contains a path variable that is not used in any other solution vector. ■

The following lemma indicates that the number of path variables in the formulation, and therefore the dimension of the corresponding polytope, can become exponentially large in terms of the size of the graph.

Lemma 6.3.1 *The number of distinct simple paths (a path without node repetition) between any pair of nodes in a complete graph on* $|V|$ *nodes equals* $\lfloor (|V| - 2)!e \rfloor$, *if* $|V| \geq 3$.

Proof. Let $u, v \in V$, then there remain $|V| - 2$ nodes that may function as intermediate nodes on a path from u to v. Since for a given number of k intermediate nodes, the number of possible orders equals $k!$, the total number of paths between nodes u and v in a complete graph equals

$$\sum_{i=0}^{|V|-2} \binom{|V|-2}{i} i! = \sum_{i=0}^{|V|-2} \frac{(|V|-2)!}{(|V|-2-i)!} = \sum_{i=0}^{|V|-2} \frac{(|V|-2)!}{i!}$$

To analyze this sum, note that

$$\sum_{i=0}^{n} \frac{n!}{i!} = n! \sum_{i=0}^{n} \frac{1}{i!} = n! \sum_{i=0}^{\infty} \frac{1}{i!} - n! \sum_{i=n+1}^{\infty} \frac{1}{i!} = n!e - n! \sum_{i=n+1}^{\infty} \frac{1}{i!}$$

Moreover, since

$$\begin{aligned} n! \sum_{i=n+1}^{\infty} \frac{1}{i!} &= \frac{n!}{(n+1)!} + \frac{n!}{(n+2)!} + \frac{n!}{(n+3)!} + \ldots \\ &< \frac{1}{n+1} + \frac{1}{(n+1)^2} + \frac{1}{(n+1)^3} + \ldots \\ &= \sum_{i=1}^{\infty} \frac{1}{(n+1)^i} = \frac{1}{n} < 1, \text{ for } n \geq 1, \end{aligned}$$

it holds that

$$n! \sum_{i=0}^{n} \frac{1}{i!} = n!e - n! \sum_{i=n+1}^{\infty} \frac{1}{i!} = \lfloor n!e \rfloor$$

This completes the proof. ■

Proposition 6.3.2 *Let $q \in Q, p \in P^q$. The trivial inequalities $z_p^q \geq 0$ are valid and facet defining for $conv(UNPM)$ and $conv(DNPM)$.*

Proof. Let $\tilde{q} \in Q$ and $\tilde{p} \in P^{\tilde{q}}$. W.l.o.g. we assume that the path $\hat{p}$ for commodity $\tilde{q}$ as chosen in the first solution vector of the proof of Proposition 6.3.1 satisfies $\hat{p} \neq \tilde{p}$. But then all of the vectors but one as constructed in the proof of Proposition 6.3.1 satisfy $z_{\tilde{p}}^{\tilde{q}} = 0$. Hence, we have identified $|E| + \sum_{q \in Q}(|P^q| - 1)$ affinely independent vectors in the convex hull of feasible solutions that satisfy the inequality at equality, which completes the proof. ■

Theorem 6.3.1 *Any non-trivial facet defining inequality for $conv(X_{ij}^{UP})$ is a non-trivial facet defining inequality for $conv(UNPM)$.*

Proof. Let $a_{ij}x_{ij} \geq \sum_{q \in Q} \sum_{p \in P_{ij}^q \cup P_{ji}^q} b_p^q z_p^q - c$ be a non-trivial facet defining inequality for $conv(X_{ij}^{UP})$. If k denotes the dimension of $conv(X_{ij}^{UP})$ then $k = 1 + \sum_{q \in Q}(|P_{ij}^q| + |P_{ji}^q|)$, since the polytope $conv(X_{ij}^{UP})$ is full dimensional. For each $q \in Q$, let $\bar{p}^q \notin P_{ij}^q \cup P_{ji}$ be a path that does not visit arc (i,j) nor (j,i). Now consider the polytope

$$T = conv\left(\left\{(x,z) \in UNPM : z_p^q = 0, \forall q \in Q, \forall p \notin (P_{ij}^q \cup P_{ji}^q \cup \{\bar{p}^q\})\right\}\right).$$

This polytope is the convex hull of the set of solutions for the restricted network loading problem where a commodity $q \in Q$ can only be routed on path $\bar{p}^q$ or on a path that visits edge $\{i,j\}$. Using Proposition 6.3.1 the dimension of T thus is $|E| + \sum_{q \in Q}(|P_{ij}^q| + |P_{ji}^q|) = k + |E| - 1$. First we show that the inequality $a_{ij}x_{ij} \geq \sum_{q \in Q} \sum_{p \in P_{ij}^q \cup P_{ji}^q} b_p^q z_p^q - c$ is also a facet defining inequality for T by constructing $k + |E| - 1$ solution vectors in T that satisfy the inequality at equality. Note that there exist k affinely independent vectors $(\bar{x}, \bar{z}) \in X_{ij}^{UP}$ that satisfy the inequality at equality. Given such a vector $(\bar{x}, \bar{z})$ we define a vector $(\tilde{x}, \tilde{z}) \in T$ as follows. For all $q \in Q$, let $\tilde{z}_p^q = \bar{z}_p^q$ for all $p \in P_{ij}^q \cup P_{ji}^q$, $\tilde{z}_{\bar{p}}^q = 1$ if

$\sum_{p \in P^q_{ij} \cup P^q_{ji}} \bar{z}^q_p = 0$, $\bar{z}^q_{\bar{p}} = 0$ otherwise, and $\bar{z}^q_p = 0$ for all $p \notin (P^q_{ij} \cup P^q_{ji} \cup \{\bar{p}^q\})$. Moreover, define $\bar{x}_{ij} = \bar{x}_{ij}$ and $\bar{x}_{uv} = \lceil \sum_{q \in Q} d^q \rceil$, for all $\{u, v\} \neq \{i, j\}$. Then these k vectors $(\bar{x}, \bar{z}) \in T$ are also affinely independent. Moreover, for any of these given vectors, we can install one additional unit of capacity on any of the edges $\{u, v\} \neq \{i, j\}$, which leads to $|E| - 1$ additional vectors. All of these $k + |E| - 1$ vectors are affinely independent and satisfy the inequality at equality, hence, the inequality is also facet defining for the polytope T.

Next, we prove that maximal sequential lifting applied to a variable that is fixed to zero in the polytope T yields a lifting coefficient zero, which implies that the inequality is also facet defining for the $conv(UNPM)$. Thus, let $\tilde{q} \in Q$ and let $\tilde{p} \notin (P^{\tilde{q}}_{ij} \cup P^{\tilde{q}}_{ji} \cup \{\bar{p}^{\tilde{q}}\})$. If we apply maximal lifting on the variable $z^{\tilde{q}}_{\tilde{p}}$ to obtain a valid inequality $a_{ij}x_{ij} \geq \sum_{q \in Q} \sum_{p \in P^q_{ij} \cup P^q_{ji}} b^q_p z^q_p + b^{\tilde{q}}_{\tilde{p}} z^{\tilde{q}}_{\tilde{p}} - c$, then the lifting coefficient $b^{\tilde{q}}_{\tilde{p}}$ is determined by

$$b^{\tilde{q}}_{\tilde{p}} = \min_{\substack{(x,z) \in UNPM : z^{\tilde{q}}_{\tilde{p}} = 1, \\ z^q_p = 0, \forall q \in Q, \forall p \notin (P^q_{ij} \cup P^q_{ji} \cup \{\bar{p}^q, \bar{p}^{\tilde{q}}\})}} \{a_{ij}x_{ij} - \sum_{q \in Q} \sum_{p \in P^q_{ij} \cup P^q_{ji}} b^q_p z^q_p + c\}.$$

Since the facet defining inequality under consideration is non-trivial, there exists a solution $(\bar{x}, \bar{z}) \in T$ with $\bar{z}^{\tilde{q}}_{\bar{p}} = 1$ that satisfies the inequality at equality. Now consider the solution vector that is obtained by replacing path $\bar{p}$ by $\tilde{p}$ for commodity q. This yields a solution vector that is feasible for the minimization lifting problem and has objective value zero since the coefficient of the variable $z^{\tilde{q}}_{\tilde{p}}$ is zero in the facet defining inequality. Since the lifting coefficient is nonnegative it then follows that it must be zero.. Repeating this argument for all remaining variables that are currently fixed to zero yields the desired result. ■

Theorem 6.3.2 *Any non-trivial facet defining inequality for $conv(Y^{DP}_{ij})$ is a non-trivial facet defining inequality for $conv(DNPM)$.*

Proof. Similar to the proof of Theorem 6.3.1. ■

6.3.2 Flow Formulation

Proposition 6.3.3 *The dimension of both $conv(UNFM)$ and $conv(DNFM)$ are equal to $|E| + |Q| * (|A| - |V| + 1)$.*

Proof. The number of edge capacity variables equals $|E|$ and the number of flow variables equals $|Q| * |A|$. Furthermore, since for each commodity there are $|V|$ flow balance constraints, of which $|V| - 1$ are linearly independent, an upper bound on the dimension is given by $|E| + |Q| * (|A| - |V| + 1)$. To prove that this bound is tight we show that there exist no other implicit equalities in the model. Stated differently, if

$$\sum_{\{i,j\} \in E} \alpha_{ij}x_{ij} + \sum_{q \in Q} \sum_{\{i,j\} \in E} (\beta_{ij} f^q_{ij} + \beta_{ji} f^q_{ji}) = \delta$$

is satisfied by each solution in $UNFM$, we prove that this equality is a linear combination of the model equalities.

Let $\{u,v\} \in E$, and let $(x,f) \in UNFM$. Next, define a solution $(\bar{x},\bar{f})$ as $\bar{f} = f$, $\bar{x}_{uv} = x_{uv} + 1$ and $\bar{x}_{ij} = x_{ij}$ for all $\{i,j\} \neq \{u,v\}$. Because both solutions satisfy the equality, it holds that $\alpha_{uv} = 0$, and since the edge was chosen arbitrarily it follows that $\alpha_{ij} = 0$ for all $\{i,j\} \in E$.

Next we show that for all $q \in Q$ and for all cycles C in the graph it holds that $\sum_{(i,j)\in C} \beta_{ij}^q = 0$. Since any cycle in the graph can be decomposed into a collection of simple cycles (i.e. cycles that visit each node at most once) it follows that we only have to prove this claim for simple cycles.

Let $\hat{q} \in Q$ and C a simple cycle in the graph. First we consider the case that C is a 2-cycle (a cycle of two arcs, say (u,v) and (v,u) for some $u,v \in V$). Since there exist two node disjoint paths from $s^{\hat{q}}$ to $t^{\hat{q}}$ in the graph, there exists a path from $s^{\hat{q}}$ to $t^{\hat{q}}$ that does not contain edge $\{u,v\}$. Let $(x,f) \in UNFM$ be a solution that uses this specific path for the routing of commodity $\hat{q}$. Given this solution, let $(\bar{x},\bar{f}) \in UNFM$ be a solution that employs exactly the same routing strategy for all commodities $q \in Q$, except that commodity $\hat{q}$ is additionally routed on arcs (u,v) and (v,u). Since both solutions satisfy the equality and $\alpha_{ij} = 0$ for all $\{i,j\} \in E$ it follows that $\beta_{uv}^{\hat{q}} + \beta_{vu}^{\hat{q}} = 0$.

Now we consider the case that C is not a 2-cycle. Let p be a simple path from $s^{\hat{q}}$ to $t^{\hat{q}}$ in the graph. If the number of nodes on the path p that are also on the cycle C is less than or equal to one, then we use similar arguments as before to show that $\sum_{(i,j)\in C} \beta_{ij}^{\hat{q}} = 0$. Let solution $(x,f) \in UNFM$ use path p for the routing of commodity $\hat{q}$. Next, define solution $(\bar{x},\bar{f})$ to be a solution that employs exactly the same routing for all commodities $q \in Q$, except that commodity $\hat{q}$ is also routed on cycle C. Comparing the two solutions, and using the fact that $\alpha_{ij} = 0$ for all $\{i,j\} \in E$, it follows that $\sum_{(i,j)\in C} \beta_{ij}^{\hat{q}} = 0$.

If the number of nodes on path p that are also on the cycle C is greater than or equal to 2, then define v_1 as the first, and v_2 to be the last node on the path that is also on the cycle. As a result, path p can be decomposed into three parts p_1, p_2, p_3, where p_1 is a path from $s^{\hat{q}}$ to v_1, p_2 is a path from v_1 to v_2, and p_3 is a path from v_2 to $t^{\hat{q}}$. Similarly, the cycle C can be decomposed into a path C_1 from v_1 to v_2 and a path C_2 from v_2 to v_1. Given these definitions, we can construct two new paths from $s^{\hat{q}}$ to $t^{\hat{q}}$ in the graph. The first path can be represented as p_1, C_1, p_3 and the second path as p_1, C_2^r, p_3, where C_2^r is the reversed path of C_2. Let $(x,f) \in UNFM$ be a solution that uses the first path for the routing of commodity $\hat{q}$. Given this solution, define a solution $(\bar{x},\bar{f}) \in UNFM$ that employs the same routing strategy for all commodities $q \in Q \setminus \{\hat{q}\}$, but uses the second path for commodity $\hat{q}$. Since both solutions satisfy the equality it follows that $\sum_{(i,j)\in C_1} \beta_{ij}^{\hat{q}} - \sum_{(i,j)\in C_2^r} \beta_{ij}^{\hat{q}} = 0$. Exploiting the fact that $\beta_{ij}^q = -\beta_{ji}^q$ for all $q \in Q$ and for all $\{i,j\} \in E$, it follows that $\sum_{(i,j)\in C} \beta_{ij}^{\hat{q}} = \sum_{(i,j)\in C_1} \beta_{ij}^{\hat{q}} + \sum_{(i,j)\in C_2} \beta_{ij}^{\hat{q}} = 0$, which proves our intermediate claim.

Next, for all $q \in Q$, for all $i \in V$ and a path p from s^q to i in the graph, let $\mu_i^q = \sum_{(i,j)\in p} \beta_{ij}^q$. We claim that the value of μ_i^q is independent of the selected path p. To verify this claim,

let p_1, p_2 be two paths from s^q to i in the graph, and let p_1^r, p_2^r be the reversed paths. Then $p_1 \cup p_2^r$ forms a cycle, hence, $\sum_{(i,j)\in p_1\cup p_2^r} \beta_{ij}^q = 0$. Using $\beta_{ij}^q = -\beta_{ji}^q$ it then follows that $\sum_{q\in p_1} \beta_{ij}^q = \sum_{q\in p_2} \beta_{ij}^q$, thus indeed, the value of μ_i^q is independent of the selected path from s^q to i.

If we multiply the flow conservation equalities of the model $UNFM$ by these multipliers and add them all up, we obtain the following expression:

$$\begin{aligned}\sum_{q\in Q}\sum_{i\in V}\mu_i^q\left(\sum_j f_{ji}^q - \sum_j f_{ij}^q\right) &= \sum_{q\in Q}\sum_{\{i,j\}\in E}\{(\mu_i^q - \mu_j^q)f_{ji}^q + (\mu_j^q - \mu_i^q)f_{ij}^q\} \\ &= \sum_{q\in Q}\sum_{\{i,j\}\in E}(\beta_{ij}^q f_{ij}^q + \beta_{ji}^q f_{ji}^q)\end{aligned}$$

This implies that the equality is indeed a linear combination of the model equalities. ■

Proposition 6.3.4 *Let $q \in Q$ and $(i,j) \in A$. The trivial inequalities $f_{ij}^q \geq 0$ are valid and facet defining for $conv(UNFM)$ and $conv(DNFM)$.*

Proof. Analogous to the proof of Proposition 6.3.3. ■

Theorem 6.3.3 *Any non-trivial facet defining inequality for $conv(X_{ij}^{UF})$ is a non-trivial facet defining inequality for $conv(UNFM)$.*

Proof. Analogous to the proof of Proposition 6.3.3. ■

Theorem 6.3.4 *Any non-trivial facet defining inequality for $conv(Y_{ij}^{DF})$ is a non-trivial facet defining inequality for $conv(DNFM)$.*

Proof. Analogous to the proof of Proposition 6.3.3. ■

6.4 Characteristics of the Edge Capacity Polytope

In Section 6.2 we introduced six different polytopes restricted to a single edge of the original model. The edge models X_{ij}^{UF}, X_{ij}^{UP}, X_{ij}^{DF}, X_{ij}^{DP} are similar. They describe a knapsack with variable integer capacity. Since the associated polyhedra are the same, in this and the following sections we use easier notation and a redefinition of the edge capacity model that captures all of the aforementioned edge models. We show that optimizing a linear function over the polytope is an NP-hard problem in general. Next, we state the dimension of the polytope, discuss its trivial facets and derive a general form of a facet defining inequality. The main result in this section is a shifting theorem, which indicates that the complete set of facet defining inequalities for the edge capacity polytope can be

obtained from a related edge capacity polytope in which the demands satisfy $d^q \in (0,1]$ for all $q \in Q$. Moreover, we provide bounds on the coefficients in a facet defining inequality, and derive necessary and sufficient conditions under which the model inequality yields the complete description of the edge capacity polytope.

Consider a set Q of items (commodities) and let $d^q \in \mathbb{Q}^+$ represent the size (demand) for an item $q \in Q$ (normalized to the base capacity λ). Let the integer variable x denote the number of capacity units selected and let the binary variables f^q indicate whether or not an individual item q is selected. The edge capacity set is then defined as

$$X = \left\{(x, f) \in \mathbb{N} \times \{0,1\}^{|Q|} : x \geq \sum_{q \in Q} d^q f^q\right\} \tag{6.19}$$

Given an arbitrary objective function $(\delta, \gamma) \in \mathbb{Z} \times \mathbb{Z}^{|Q|}$, the problem of minimizing the objective over the set X is NP-hard in general. To formalize this statement, we define the following decision problems.

EDGE CAPACITY PROBLEM
INSTANCE: Set of items Q, demand size $d^q \in \mathbb{N}$, for all $q \in Q$, objective coefficients $\gamma^q \in \mathbb{Z}$, for all $q \in Q$, objective coefficient $\delta \in \mathbb{Z}$, capacity coefficient $\lambda \in \mathbb{N}$, and an integer $K \in \mathbb{Z}$.
QUESTION: Does there exist a vector $(x, f) \in \mathbb{N} \times \{0,1\}^{|Q|}$ such that $\lambda x \geq \sum_{q \in Q} d^q f^q$ and $\delta x + \sum_{q \in Q} \gamma^q f^q \leq K$?

PARTITION (see Garey and Johnson [37])
INSTANCE: Set of items A, size $s^q \in \mathbb{N}$, for all $q \in A$, with $\sum_{q \in A} s^q$ even.
QUESTION: Does there exist a subset $\bar{A} \subseteq A$ such that $\sum_{q \in \bar{A}} s^q = \sum_{q \in A \setminus \bar{A}} s^q$?

Theorem 6.4.1 EDGE CAPACITY PROBLEM *is NP-complete.*

Proof. It is easy to verify that EDGE CAPACITY PROBLEM is in NP. Next, we show that PARTITION reduces to EDGE CAPACITY PROBLEM. Given an instance of PARTITION, define an instance of EDGE CAPACITY PROBLEM as follows. Let $Q = A \cup \{\hat{q}\}$, $d^q = s^q$, for all $q \in A$, and $d^{\hat{q}} = 1$. Next define $\gamma^q = -d^q$, for all $q \in A$, and $\gamma^{\hat{q}} = -2$. Finally, let $\lambda = \frac{1}{2}\sum_{q \in A} s^q + 1$, $\delta = \lambda$ and $K = -1$. This is obviously a polynomial transformation. Next it remains to show that an instance of PARTITION yields an affirmative answer if and only if the corresponding instance of EDGE CAPACITY PROBLEM yields an affirmative answer. If $\bar{A} \subseteq A$ is a set of items satisfying $\sum_{q \in \bar{A}} s^q = \sum_{q \in A \setminus \bar{A}} s^q$, then the solution (x, f) defined as $x = 1$, $f^q = 1, q \in \bar{A} \cup \{\hat{q}\}$, $f^q = 0$ otherwise, gives a vector satisfying the required conditions. Conversely, if there exists a vector (x, f) which satisfies $\lambda x \geq \sum_{q \in Q} d^q f^q$ and $\delta x + \sum_{q \in Q} \gamma^q f^q \leq -1$, then it is easy to see that $x > 0$, since $x = 0$ would imply that $f^q = 0$ for all $q \in Q$, a contradiction with the fact that $\delta x + \gamma^T f$ has negative value. Moreover, if $x \geq 2$, then $\delta x + \gamma^T f \geq 2\lambda + \gamma^T f \geq 2\lambda - \sum_{q \in Q} \gamma^q = \sum_{q \in A} s^q + 2 - \sum_{q \in A} s^q - 2 = 0$. Hence, $x = 1$. Next, from $\delta x + \gamma^T f \leq -1$ it follows that $\gamma^T f \leq -\lambda - 1$ which implies $|\gamma^T f| \geq \lambda + 1$. From $\lambda x \geq d^T f$ it follows that $\sum_{q \in Q} d^q f^q \leq \lambda$. Since for each element in

A it holds that $d^q = -\gamma^q$, the only way that $|\gamma^T f|$ can be greater than $\sum_{q\in Q} d^q f^q$ is if $f^{\bar{q}} = 1$. But then it follows that both $\sum_{q\in A} \gamma^q f^q \leq -\lambda + 1$ and $\sum_{q\in A} d^q f^q \leq \lambda - 1$ which is equivalent to $\sum_{q\in A} d^q f^q = \lambda - 1 = \frac{1}{2}\sum_{q\in A} s^q$. Hence, let $\bar{A} = \{q \in A : f^q = 1\}$, then $\bar{A}$ is the required subset. ■

Let $D^q = \lceil d^q \rceil$ be the smallest integer greater than or equal to d^q, and let $r^q = d^q + 1 - D^q \in (0, 1]$ be the 'fractional' part of the demand. Similarly, for a subset $S \subseteq Q$, let $d(S) = \sum_{q\in S} d^q$, $D(S) = \lceil d(S) \rceil$ and $r(S) = d(S) + 1 - D(S)$. Finally, define $e^S \in \{0,1\}^Q$ as the characteristic vector of a set S, i.e. $e^S_i = 1$ if $i \in S$ and zero otherwise.

Lemma 6.4.1 *The dimension of conv(X) is $|Q| + 1$.*

Proof. The vectors $(x, f) = (0, 0)$, $(x, f) = (1, 0)$ and $(x, f) = (D^q, e^{\{q\}})$ for all $q \in Q$ yield $|Q| + 2$ affinely independent vectors in X. ■

Proposition 6.4.1 *For all $q \in Q$, the trivial inequalities $f^q \geq 0$ and $f^q \leq 1$ define facets of conv(X).*

Proof. For $f^q \geq 0$, $|Q| + 1$ affinely independent solutions $(x, f) \in X$ are given by $(0, 0)$, $(1, 0)$ and (D^i, e^i) for all $i \in Q \setminus \{q\}$. For $f^q \leq 1$, $|Q| + 1$ affinely independent solutions $(x, f) \in X$ are given by (D^q, e^q), $(D^q + 1, e^q)$ and $(D(\{i, q\}), e^{\{i,q\}})$ for all $i \in Q \setminus \{q\}$. ■

Proposition 6.4.2 *(cf. [20]) Each non-trivial facet of conv(X) can be written in the form $ax \geq \sum_{q\in Q} b^q f^q - c$, with $a, c \in \mathbb{N}, b^q \in \mathbb{N}$, for all $q \in Q$.*

Proof. A non-trivial facet defining inequality is of the form $ax \geq \sum_{q\in Q} b^q f^q - c$. The fact that $c \geq 0$ follows from the fact that the all zero solution $(x, f) = (0, 0)$ does not satisfy the inequality if $c < 0$. Next, since for any solution $(x, f) \in X$ we can increase the value of the capacity variable to an arbitrarily large integer number without violating its feasibility, it follows that $a < 0$ cannot be the case. Furthermore, if $a = 0$ then the resulting inequality $\sum_{q\in Q} b^q f^q \leq c$ is a linear combination of the trivial inequalities and the inequality $0 \leq 1$. Hence, $a > 0$. Now suppose there exists a $q' \in Q$ with $b^{q'} < 0$. Since the facet defining inequality is non-trivial there exists a solution $(\bar{x}, \bar{f}) \in X$ with $\bar{f}^{q'} = 1$ that satisfies the inequality at equality, $a\bar{x} = \sum_{q\in Q} b^q \bar{f}^q - c$. Define the solution $(\tilde{x}, \tilde{f})$ as $\tilde{x} = \bar{x}$, $\tilde{f}^{q'} = 0$, $\tilde{f}^q = \bar{f}^q$ for all $q \in Q \setminus \{q'\}$. Then $(\tilde{x}, \tilde{f}) \in X$ but $a\tilde{x} = a\bar{x} = \sum_{q\in Q} b^q \bar{f}^q - c = b^{q'}\bar{f}^{q'} + \sum_{q\in Q\setminus\{q'\}} b^q \bar{f}^q - c < b^{q'}\tilde{f}^{q'} + \sum_{q\in Q\setminus\{q'\}} b^q \tilde{f}^q - c = \sum_{q\in Q} b^q \tilde{f}^q - c$ which is in contradiction with the validity of the inequality. Hence, $b^q \geq 0$ for all $q \in Q$. Finally, the fact that any non-trivial facet defining inequality has rational coefficients (and can therefore be written with integer coefficients) follows from the fact that all extreme points of the set are rational. ■

Theorem 6.4.2 *(Shifting Theorem)*
Let $i \in Q$ and $\mu \in \mathbb{Z}$ such that $d^i + \mu \geq 0$, and define

$$X(i,\mu) = \left\{(x,f) \in \mathbb{N} \times \{0,1\}^{|Q|} : x \geq (d^i + \mu)f^i + \textstyle\sum_{q \in Q \setminus \{i\}} d^q f^q \right\} \tag{6.20}$$

Then the inequality

$$x \geq \textstyle\sum_{q \in Q} b^q f^q - c \tag{6.21}$$

is a non-trivial facet defining inequality for $conv(X)$ if and only if

$$x \geq (b^i + \mu)f^i + \textstyle\sum_{q \in Q \setminus \{i\}} b^q f^q - c \tag{6.22}$$

is a non-trivial facet defining inequality for $conv(X(i,\mu))$.

Proof. Note that we only need to prove that any facet defining inequality for $conv(X)$ can be converted as indicated to a facet defining inequality for $conv(X(i,\mu))$, since the converse then directly follows for a suitable choice $\mu' = -\mu$. First, we prove validity. Let $(\bar{x}, \bar{f}) \in X(i,\mu)$, then $(\bar{x} - \mu\bar{f}^i, \bar{f}) \in X$, hence $\bar{x} - \mu\bar{f}^i \geq \sum_{q \in Q} b^q f^q - c$ which implies that $\bar{x} \geq (b^i + \mu)f^i + \sum_{q \in Q \setminus \{i\}} b^q f^q - c$. Next, let $(x_1, f_1), \ldots, (x_{|Q|+1}, f_{|Q|+1})$ be $|Q| + 1$ affinely independent solutions of X that satisfy (6.21) at equality. Then $(x_1 + \mu f_1^i, f_1), \ldots, (x_{|Q|+1} + \mu f_{|Q|+1}^i, f_{|Q|+1})$ satisfy (6.22) at equality, and they are also affinely independent. ∎

Theorem 6.4.3 *Let $ax \geq \sum_{q \in Q} b^q f^q - c$ be a non-trivial facet defining valid inequality for $conv(X)$ with $a, c \in \mathbb{N}$, and $b^q \in \mathbb{N}$ for all $q \in Q$. Then $b^q = ad^q$ if d^q is integer and $b^q \in \{a(D^q - 1), \ldots, aD^q\}$ if d^q is not integer, for all $q \in Q$.*

Proof. Let $\bar{q} \in Q$ and let $(x_0, f_0) \in X$ be a solution with $f_0^{\bar{q}} = 0$ that satisfies the facet defining inequality at equality. Moreover, let $Q_0 = \{q \in Q : f_0^q = 1\}$. Then $aD(Q_0) = \sum_{q \in Q_0} b^q - c$ and since $\bar{q} \notin Q_0$ it follows from validity that $aD(Q_0 \cup \{\bar{q}\}) \geq (\sum_{q \in Q_0} b^q - c) + b^{\bar{q}} = aD(Q_0) + b^{\bar{q}}$. This yields an upper bound on $b^{\bar{q}}$ since we conclude $b^{\bar{q}} \leq aD(Q_0 \cup \{\bar{q}\}) - aD(Q_0)$.

Similarly, let $(x_1, f_1) \in X$ be a solution with $f_1^{\bar{q}} = 1$ that satisfies the facet defining inequality at equality, and define $Q_1 = \{q \in Q : f_1^q = 1\}$. Again, $aD(Q_1) = \sum_{q \in Q_1} b^q - c$ and since $\bar{q} \in Q_1$ it follows from validity that $aD(Q_1 \setminus \{\bar{q}\}) \geq (\sum_{q \in Q_1} b^q - c) - b^{\bar{q}} = aD(Q_1) - b^{\bar{q}}$. This implies a lower bound on $b^{\bar{q}}$, namely $b^{\bar{q}} \geq aD(Q_1) - aD(Q_1 \setminus \{\bar{q}\})$.

If $d^{\bar{q}}$ is integer, then both the lower and upper bound are equal to $ad^{\bar{q}}$ which proves the first part of our claim. If $d^{\bar{q}}$ is not integer, then $b^{\bar{q}} \leq aD(Q_0 \cup \{\bar{q}\}) - aD(Q_0) \leq aD^{\bar{q}}$ and $b^{\bar{q}} \geq aD(Q_1) - aD(Q_1 \setminus \{\bar{q}\}) \geq a(D^{\bar{q}} - 1)$. ∎

Theorem 6.4.4 *The model inequality is the unique facet defining inequality for the polyhedron $conv(X)$ if and only if $d^q \in \mathbb{N}$, for all $q \in Q$.*

Proof. If $d^{\hat{q}} \notin \mathbb{N}$, for some $\hat{q} \in Q$, then the fractional solution $(x, f) = (d^q, e^{\{\hat{q}\}})$ is an extreme point of the LP-relaxation that satisfies all the model inequalities. If $d^q \in \mathbb{N}$ for all $q \in Q$, then by Theorem 6.4.3 we have that in every inequality $x \geq \sum_{q \in Q} b^q f^q - c$ that defines a non-trivial facet of $conv(X)$, $b^q = d^q$ for all $q \in Q$. If $c > 0$, then the resulting inequality is dominated by the model inequality. Hence, the model inequality $x \geq \sum_{q \in Q} d^q f^q$ defines the only non-trivial facet in this case. ■

6.5 Lower Convex Envelop Inequalities

Using the results of Section 6.4, the analysis of the edge capacity polytope can be greatly facilitated. By Theorem 6.4.3, commodities with an integer demand can be dealt with easily given a valid inequality on the remaining commodities. Furthermore, for the remaining commodities one can concentrate on the fractional part of the demand of a commodity, i.e. first assume all demands $d^q \in (0, 1)$, generate facets, and use Theorem 6.4.2 to obtain facets for the problem with the actual demand sizes. Stated differently, one could view the demand value as the sum of two parts, namely the integral part $D^q - 1$ and the 'fractional part' $r^q \in (0, 1]$. Likewise, the value of a coefficient of a commodity in a facet defining inequality can be viewed as the sum of two parts. Theorem 6.4.2 explains the part of the coefficient that originates from the integral part $D^q - 1$ of a commodity's demand. The other part of the coefficient in a facet defining inequality which needs to be explained stems from the "fractional" part of a commodity's demand, that is r^q. Much of the analysis in this section therefore considers demand values $d^q = r^q \in (0, 1]$ for all $q \in Q$. Still, all propositions and theorems are stated such that the results are also valid for arbitrary demand values d^q.

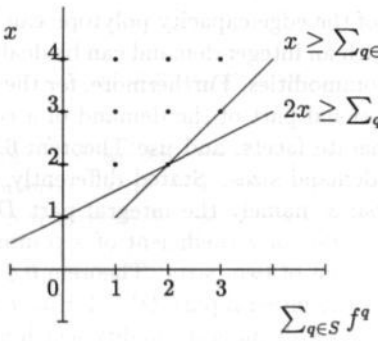
Since the values r^q are somewhat comparable in size for the different $q \in Q$ ($r^q \in (0, 1]$ for all q), one could expect this second part of a commodity's coefficient in a facet defining inequality to be somewhat comparable in size as well. In this section we therefore introduce a class of valid inequalities called lower convex envelop inequalities, which are based on this idea. This class of valid inequalities is defined on a projection of the set X. We show two different types of facet defining inequalities that may arise in the class of lower convex envelop inequalities. Moreover, we show that lifting lower convex envelop inequalities to obtain valid inequalities for X itself can be performed in polynomial time. We start with the definition of a projection of the edge capacity polytope.

Definition 6.5.1 *Let $Q^0, Q^1 \subset Q$ be disjoint subsets of Q. Then $X(Q^0, Q^1)$ defined by*

$$X(Q^0, Q^1) = \{(x, f) \in X : f^q = 0 \ \forall q \in Q^0, f^q = 1 \ \forall q \in Q^1\}$$

is the projection of X on the space with $f^q = 0$ for all $q \in Q^0$ and $f^q = 1$ for all $q \in Q^1$.

This projected edge set can be seen as a set of vectors in $(|Q|+1)$-dimensional space. Instead of representing the set $X(Q^0, Q^1)$ in $(|Q|+1)$-dimensional space, one can also plot all vectors in $X(Q^0, Q^1)$ in two-dimensional space, as in the example Figure 6.1. Let $S = Q \setminus (Q^0 \cup Q^1)$. The horizontal axis of this figure measures $\sum_{q \in S} f^q$ and the vertical axis measures the value of the capacity variable x. Hence, a solution $(x, f) \in X(Q^0, Q^1)$ is represented by a point with coordinates $(\sum_{q \in S} f^q, x)$ in the two-dimensional figure. Similarly as in $(|Q|+1)$-dimensional space, an inequality $ax \geq b \sum_{q \in S} f^q - c$ that is satisfied by all solutions in this two-dimensional space yields a valid inequality for the set $X(Q^0, Q^1)$. Note that these inequalities have the same coefficient for all $q \in S$ in the inequality. We consider the strongest possible valid inequalities arising from the two-dimensional space. These inequalities describe the lower convex envelop of the set of solutions in $X(Q^0, Q^1)$, as indicated in Figure 6.1.



$Q = \{1, 2, 3, 4\}$

$d^q = 0.6$ for all $q \in Q$

$Q^0 = \emptyset, Q^1 = \{4\}$

$S = \{1, 2, 3\}$

Figure 6.1: Lower Convex Envelop Inequalities

Assume that the commodities in S are ordered such that $d^1 \leq d^2 \leq \ldots \leq d^{|S|}$, and for $k = 1, \ldots, |S|$, let $S_k = \{1, \ldots, k\}$. Then the set of lower convex envelop inequalities basically describes the lower convex envelop of the points $(k, D(Q^1 \cup S_k))$ for $k = 1, \ldots, |S|$ in two-dimensional space. The following proposition states bounds on the slope of a lower convex envelop inequality.

Proposition 6.5.1 *If $ax \geq b \sum_{q \in S} f^q - c$ is a lower convex envelop inequality for the set $X(Q^0, Q^1)$ with demand values $d^q = r^q \in (0, 1]$ for all $q \in Q$, then the slope b/a of the lower convex envelop inequality satisfies $0 \leq b/a \leq 1$.*

Proof. Consider a lower convex envelop inequality $ax \geq b \sum_{q \in S} f^q - c$ for $X(Q^0, Q^1)$. The slope b/a of the lower convex envelop inequality is determined by two distinct points in the two-dimensional picture that satisfy the inequality at equality. Let $(k_1, D(Q^1 \cup S_{k_1}))$ and $(k_2, D(Q^1 \cup S_{k_2}))$ be two such points with $k_1 < k_2$. Then $b/a = (D(Q^1 \cup S_{k_2}) - D(Q^1 \cup S_{k_1}))/(k_2 - k_1)$. Since $S_{k_1} \subseteq S_{k_2}$ it follows that the numerator is nonnegative, which together with $k_1 \leq k_2$ implies that $b/a \geq 0$. Furthermore, $S_{k_1} \subseteq S_{k_2}$ together with

$d^q \in (0,1]$ for all $q \in S$ implies that $D(Q^1 \cup S_{k_2}) - D(Q^1 \cup S_{k_1}) \leq k_2 - k_1$. Hence, $b/a \leq 1$. ■

Next we derive necessary and sufficient conditions under which there exists a lower convex envelop inequality $ax \geq b\sum_{q\in S} f^q - c$ with slope $b/a = 1$ that is also facet defining for $conv(X(Q^0, Q^1))$. We use the following definition.

Definition 6.5.2 *Let $Q^1 \subset Q$ and $S \subset Q$ with $S \cap Q^1 = \emptyset$. Then*

$$n(Q^1, S) = \sum_{q\in S} D^q + D(Q^1) - D(S \cup Q^1) \tag{6.23}$$

In words, $n(Q^1, S)$ measures the surplus in capacity for the set of items $Q^1 \cup S$ if $D(Q^1)$ units of capacity have already been 'installed' for the set of items Q^1 and an additional D^q units of capacity are installed for each item $q \in S$. Note that the value of $n(Q^1, S)$ does not change if demand data $r^q \in (0,1]$ are considered instead of the real demand data d^q, for all $q \in Q$.

Theorem 6.5.1 *If there exists a lower convex envelop inequality $ax \geq b\sum_{q\in S} f^q - c$ with slope $b/a = 1$ for the edge capacity set with demand values $d^q = r^q \in (0,1]$, then the inequality*

$$x \geq \sum_{q\in S} D^q f^q + D(Q^1) - n(Q^1, S) \tag{6.24}$$

is valid for $conv(X(Q^0, Q^1))$ with arbitrary demand data d^q for all $q \in Q$. Moreover, this is a facet defining lower convex envelop inequality for $conv(X(Q^0, Q^1))$ if and only if $D(Q^1 \cup S \setminus \{q\}) = D(Q^1 \cup S) - D^q$ for all $q \in S$.

Proof. First consider $d^q = r^q \in (0,1]$ for all $q \in S$. If there exists a lower convex envelop inequality $ax \geq \sum_{q\in S} f^q - c$ with slope $b/a = 1$, then the point $(|S|, D(Q^1 \cup S))$ satisfies that inequality at equality. Hence, for $a = b = 1$ it follows that $c = \sum_{q\in S} f^q - D(Q^1 \cup S) = \sum_{q\in S} D^q + D(Q^1) - D(Q^1) - D(Q^1 \cup S) = n(Q^1, S) - D(Q^1)$. This yields the lower convex envelop inequality $x \geq \sum_{q\in S} f^q + D(Q^1) - n(Q^1, S)$ for demand values $d^q \in (0,1]$, for all $q \in S$. Applying Shifting Theorem 6.4.2 for elements $q \in S$ to obtain a valid inequality for the real demand data yields the desired inequality (6.24).

Now we prove that the remaining conditions are sufficient to guarantee that the inequality is facet defining for $conv(X(Q^0, Q^1))$, which has dimension $|S| + 1$. This follows from the fact that the vectors $(D(Q^1 \cup S), e^{Q^1 \cup S})$ and $(D(Q^1 \cup S \setminus \{q\}), e^{Q^1 \cup S \setminus \{q\}})$ for all $q \in S$ yield $|S| + 1$ affinely independent vectors in $X(Q^0, Q^1)$ that satisfy that inequality at equality.

Conversely, let $\hat{q} \in S$. If $D(Q^1 \cup S \setminus \{\hat{q}\}) \neq D(Q^1 \cup S) - D^{\hat{q}}$ then the solution $(x, f) = (D(Q^1 \cup S \setminus \{\hat{q}\}), e^{Q^1 \cup S \setminus \{\hat{q}\}})$ is not on the face of the valid inequality $x \geq \sum_{q\in S} D^q f^q + D(Q^1) - n(Q^1, S)$. For any set $T \subseteq S \setminus \{\hat{q}\}$ it holds that $D(Q^1 \cup S \setminus \{\hat{q}\}) \leq D(Q^1 \cup T) + \sum_{q\in S\setminus(T\cup\{\hat{q}\})} D^q$. Hence, the vector $(D(Q^1 \cup T), e^{Q^1 \cup T})$ does not satisfy the inequality

at equality, since $D(Q^1 \cup T) \geq D(Q^1 \cup S \setminus \{\hat{q}\}) - \sum_{q \in S \setminus (T \cup \{\hat{q}\})} D^q > \sum_{q \in S \setminus \{\hat{q}\}} D^q - c - \sum_{q \in S \setminus (T \cup \{\hat{q}\})} D^q = \sum_{q \in T} D^q - c$. Thus there exists no solution $(x, f) \in X(Q^0, Q^1)$ with $f^{\hat{q}} = 0$ that satisfies the inequality at equality. As a result, the face of the inequality is a subset of the face defined by the inequality $f^{\hat{q}} \leq 1$, which implies that the inequality is not facet defining. ■

Under the conditions of the above theorem, there exists a facet defining lower convex envelop inequality with slope equal to one for $conv(X(Q^0, Q^1))$. In this case one can also identify conditions such that there exists a second facet defining lower convex envelop inequality for this polytope. This is the subject of the following theorem.

Theorem 6.5.2 *Consider $X(Q^0, Q^1)$ for demand data $d^q = r^q \in (0, 1]$. Assume that $D(Q^1 \cup S \setminus \{q\}) = D(Q^1 \cup S) - D^q$ for all $q \in S$ such that there exists a lower convex envelop inequality with slope equal to one. If $D(Q^1 \cup S_{|S|-1}) = D(Q^1 \cup S_{|S|-2})$, then*

(i). there exists a second lower convex envelop inequality $\alpha x \geq \sum_{q \in S} f^q + \delta$ through point $(|S| - 1, D(Q^1 \cup S_{|S|-1}))$ in the two-dimensional figure for demand data $d^q = r^q \in (0, 1]$.

(ii). the slope $1/\alpha$ of this lower convex envelop inequality equals

$$1/\alpha \;=\; \max_{k=0,\ldots,|S|-2} \left\{ \frac{D(Q^1 \cup S) - 1 - D(Q^1 \cup S_k)}{|S| - 1 - k} \right\}$$

for demand data $d^q = r^q \in (0, 1]$ for all $q \in S$.

(iii). the constant δ of this lower convex envelop inequality equals $\delta = \alpha D(Q^1 \cup S) - (|S| + \alpha - 1)$

(iv). the inequality

$$\alpha x \geq \sum_{q \in S} \{\alpha(D^q - 1) + 1\} f^q + \alpha D(Q^1 \cup S) - (|S| + \alpha - 1) \tag{6.25}$$

is a facet defining lower convex envelop inequality for $conv(X(Q^0, Q^1))$ with arbitrary demand data d^q for all $q \in Q$.

Proof. The lower convex envelop inequality as discussed in Theorem 6.5.1 is the most right lower convex envelop inequality of the lower convex envelop, with slope equal to one. This first inequality is guaranteed to exists due to the conditions of the theorem and the point $(|S|-1, D(Q^1 \cup S_{|S|-1})) = (|S|-1, D(Q^1 \cup S) - 1)$ is on the corresponding line. Because of the second condition $D(Q^1 \cup S_{|S|-1}) = D(Q^1 \cup S_{|S|-2})$, the point $(|S| - 2, D(Q^1 \cup S_{|S|-2}))$ is not on this line. Hence, there exists a second lower convex envelop inequality $\alpha x \geq \sum_{q \in S} f^q + \delta$ through the point $(|S|-1, D(Q^1 \cup S_{|S|-1}))$. Apart from this point, there exists a second point $(k, D(Q^1 \cup S_k))$ on this second lower convex envelop inequality, for some $k \in \{0, \ldots, |S| - 2\}$. Since the lower convex envelop inequality must be valid, it follows that that its slope is equal to the maximal slope between point $(|S| - 1, D(Q^1 \cup S) - 1)$ and $(k, D(Q^1 \cup S_k))$, over all $k \in \{0, \ldots, |S| - 2\}$.

Since the point $(|S|-1, D(Q^1 \cup S_{|S|-1}))$ is on the line of this new lower convex envelop inequality and $d^q = r^q \in (0,1]$ for all $q \in S$, it holds that $\delta = \alpha D(Q^1 \cup S_{|S|-1}) - \sum_{q \in S_{|S|-1}} f^q = \alpha(D(Q^1 \cup S) - 1) - |S| + 1 = \alpha D(Q^1 \cup S) - (|S| + \alpha - 1)$. Finally, if we apply Shifting Theorem 6.4.2 to the items $q \in S$ then the desired inequality is obtained. The fact that this is facet defining for $conv(X(Q^0, Q^1))$ follows from the fact that the solution vectors $(D(Q^1 \cup S \setminus \{q\}), e^{Q^1 \cup S \setminus \{q\}})$ for all $q \in S$ all satisfy the inequality at equality. Moreover, the vector $(x, f) = (D(Q^1 \cup S_k), e^{Q^1 \cup S_k})$ for the value of $k \in \{0, \ldots, |S| - 2\}$, such that the point $(k, D(Q^1 \cup S_k))$ is also on the line, also satisfies the inequality at equality. This yields $|S| + 1$ affinely independent vectors in $X(Q^0, Q^1)$ on the face of the inequality, hence it is facet defining. ■

Theorem 6.5.3 *Any lower convex envelop inequality for $conv(X(Q^0, Q^1))$ with demand data $d^q = r^q \in (0,1]$ for all $q \in S$ can be written in the form $ax \geq b \sum_{q \in S} f^q - c$ for certain $a, b, c \in \mathbb{N}$, and with $b \leq a \leq |S|$). Maximal lifting of this inequality to obtain a valid inequality for X can be done in polynomial time.*

Proof. Each lower convex envelop inequality is defined by two points, say $(k_1, D(Q^1 \cup S_{k_1}))$ and $(k_2, D(Q^1 \cup S_{k_2}))$ in the two-dimensional figure, for some $k_1, k_2 \in \{0, \ldots, |S|\}$, with $k_1 < k_2$. The slope b/a of such a line is then the quotient of $D(Q^1 \cup S_{k_2}) - D(Q^1 \cup S_{k_1})$ and $k_2 - k_1$, and both the numerator and denominator are bounded by $|S|$. Hence, the inequality can be written in a form with $a, b, c \in \mathbb{N}$ and $a \leq |S|$. The fact that $b \leq a$ is already shown in Proposition 6.5.1.

Using inductive arguments, one can show that the lifting coefficients b^q obtained by maximal sequential lifting of variables in $q \in Q^0 \cup Q^1$ also satisfy $b^q \leq a \leq |S|$. Now consider the lifting process for a variable $f^{\hat{q}}, \hat{q} \in Q^0$, then the value of the lifting coefficient $b^{\hat{q}}$ is given by

$$b^{\hat{q}} = \min_{(x,f) \in X(Q^0 \setminus \{\hat{q}\}, Q^1) : f^{\hat{q}} = 1} \{ax - b \sum_{q \in S} f^q + c\}$$

Note that the optimal value of the variable x in this minimization problem is bounded by $|Q|$. Moreover, for a fixed value of x, the problem simplifies to a knapsack problem. Therefore, the problem can be solved by computing at most $|Q|$ knapsack problems. Moreover, each of these knapsack problems has an objective function in which each coefficient b is bounded by $|S|$. This implies that an upper bound on the optimal value of each individual knapsack problem is given by $\mathcal{O}(|Q|^2)$, hence, they can be solved in $\mathcal{O}(|Q|^3)$ time. The lifting coefficient can thus be determined in $\mathcal{O}(|Q|^4)$ time, since it involves at most $|Q|$ knapsack problems. A similar argument holds for the lifting of a variable in Q^1. Since the total number of variables to be lifted is bounded by $|Q|$, the complete lifting process can be performed in $\mathcal{O}(|Q|^5)$ time. ■

6.6 Integer Lifting of Knapsack Inequalities

In this section we state a different approach to obtain the two types of valid lower convex envelop inequalities for the edge capacity polytope as mentioned in the previous section. We show that they can be obtained from valid inequalities for a related 0–1 knapsack polytope by integer lifting techniques. Two observations can be made regarding this integer lifting procedure. First of all, the computation of these inequalities, i.e. the computation of the lifting coefficients is NP-hard in general. Secondly, any valid inequality for the knapsack polytope will in general lead to zero, one or two valid inequalities for the edge capacity polytope. For the special case of valid cover inequalities for the knapsack polytope, we show that the lifting process can be done in polynomial time. Moreover, at least one valid inequality for the edge capacity polytope will be obtained from each cover inequality. Finally, we identify necessary and sufficient conditions under which two distinct valid inequalities for the edge capacity polytope are obtained from a single cover inequality for the 0–1 knapsack polytope.

Definition 6.6.1 *Let Q^0, Q^1, S be a partition of Q, and let $\bar{x} \in \mathbb{N}$ with $\bar{x} \geq D(Q^1 \cup \{q\})$ for all $q \in S$. Let $b = \bar{x} - d(Q^1)$, then $X(Q^0, Q^1, \bar{x})$ is a knapsack set defined by*

$$\begin{aligned} X(Q^0, Q^1, \bar{x}) &= \{(x, f) \in X : f^q = 0 \ \forall q \in Q^0, f^q = 1 \ \forall q \in Q^1, x = \bar{x}\} \\ &\simeq \{f \in \{0,1\}^{|S|} : \textstyle\sum_{q \in S} d^q f^q \leq b\} \end{aligned}$$

$X(Q^0, Q^1, \bar{x})$ is the projection of X on the space with $f^q = 0$ for all $q \in Q^0$, $f^q = 1$ for all $q \in Q^1$, and $x = \bar{x}$. Note that the condition $\bar{x} \geq D(Q^1 \cup \{q\})$ for all $q \in S$ implies that $dim(conv(X(Q^0, Q^1, \bar{x}))) = |S|$, that is, the knapsack polytope is fully dimensional.

Theorem 6.6.1 *(cf. [78]) Let Q^0, Q^1, S be a partition of the set Q, and let $\bar{x} \in \mathbb{N}$ be an integer with $\bar{x} \geq D(Q^1 \cup \{q\})$ for all $q \in S$. If $\sum_{q \in S} \pi^q f^q \leq \pi^0$ is a valid inequality for $X(Q^0, Q^1, \bar{x})$ then $\sum_{q \in S} \pi^q f^q \leq \pi^0 + \alpha(x - \bar{x})$ is a valid inequality for $X(Q^0, Q^1)$ if and only if*

$$\alpha^L := \max_{x \in \mathbb{N}: x > \bar{x}} \left\{ \frac{\eta(x) - \pi^0}{x - \bar{x}} \right\} \leq \alpha \leq \min_{x \in \mathbb{N}: D(Q^1) \leq x < \bar{x}} \left\{ \frac{\pi^0 - \eta(x)}{\bar{x} - x} \right\} =: \alpha^U$$

where

$$\eta(x) = \max \left\{ \textstyle\sum_{q \in S} \pi^q f^q : \sum_{q \in S} d^q f^q \leq x - d(Q^1), f^q \in \{0,1\}, \forall q \in S \right\}.$$

Moreover, if $\sum_{q \in S} \pi^q f^q \leq \pi^0$ is a facet defining inequality for $conv(X(Q^0, Q^1, \bar{x}))$ and $\alpha^L \leq \alpha^U$, then $\sum_{q \in S} \pi^q f^q \leq \pi^0 + \alpha^L(x - \bar{x})$ and $\sum_{q \in S} \pi^q f^q \leq \pi^0 + \alpha^U(x - \bar{x})$ are facet defining inequalities for $conv(X(Q^0, Q^1))$.

To determine α^L and α^U we can solve the knapsack problem $\eta(x^U)$ by dynamic programming where x^U is an appropriately chosen upper bound on the value of the capacity variable x, for instance $x^U = D(Q^1 \cup S)$. In general, $\alpha^L \leq \alpha^U$ does not necessarily hold, in which case integer lifting is not possible. Next we describe the main result of this section.

Theorem 6.6.2 *Let Q^0, Q^1, S be a partition of the set Q such that $S \neq \emptyset$, and let $\bar{x} \in \mathbb{N}$ satisfy $\bar{x} \geq D(Q^1 \cup \{q\})$ for all $q \in S$. If S is a minimal cover for the knapsack polytope $X(Q^0, Q^1, \bar{x})$, then*

(i). integer lifting of the minimal cover inequality can be done in polynomial time,

(ii). if $d^q \in (0,1]$ for all $q \in S$, then $1 = \alpha^L \leq \alpha^U$,

(iii). if $d^q \in (0,1]$ for all $q \in Q$, then $\alpha^U > 1$ if and only if $D(Q^1 \cup S_{|S|-1}) = D(Q^1 \cup S_{|S|-2})$,

(iv). the resulting facet defining inequalities for $conv(X(Q^0, Q^1))$ are:

$$x \geq \sum_{q \in S} D^q f^q + D(Q^1) - n(Q^1, S) \tag{6.26}$$

$$\alpha^U x \geq \sum_{q \in S} \{\alpha^U (D^q - 1) + 1\} f^q + \alpha^U D(Q^1 \cup S) - (|S| + \alpha^U - 1). \tag{6.27}$$

Proof. The knapsack problem that needs to be solved in order to determine the lifting coefficients has the same objective coefficients for all items. Hence, a sorting algorithm can solve the knapsack problem in polynomial time. To prove the remainder of the theorem, assume that $d^q \in (0,1]$ for all $q \in S$. Since S is a minimal cover, $\sum_{q \in S \setminus \{i\}} d^q \leq \bar{x} - d(Q^1)$ for all $i \in S$. Hence, from $d^q \in (0,1]$ for all $q \in Q$ it follows that in the special case of a minimal cover inequality $\eta(x) = |S|$ for all $x > \bar{x}$, and hence, the maximum value for α^L is attained for $x = \bar{x} + 1$, which yields $\alpha^L = 1$. For $x < \bar{x}$, it is easy to see that $\eta(x) \leq \eta(x+1) - 1$. Therefore, for $x < \bar{x}$ it holds that $(\pi^0 - \eta(x))/(\bar{x} - x) \geq (\bar{x} - x)/(\bar{x} - x) = 1$, such that $\alpha^U \geq \alpha^L$.

Next, if $D(Q^1 \cup S_{|S|-1}) > D(Q^1 \cup S_{|S|-2})$ then there exist $i, j \in S$ such that $n(Q^1, S \setminus \{i,j\}) = n(Q^1, S \setminus \{i\})$, which implies $\eta(\bar{x} - 1) = |S| - 2$, and hence $\alpha^U \leq 1$. Together with $\alpha^U \geq 1$ this yields $\alpha^U = 1$. Conversely, if $n(Q^1, S \setminus \{i,j\}) \neq n(Q^1, S \setminus \{i\})$ for all $i, j \in S$, then $\eta(\bar{x} - 1) \leq |S| - 3$. Hence, for $x = \bar{x} - 1$ the quotient $(\pi^0 - \eta(x))/(\bar{x} - x)$ is strictly greater than 1. Moreover, again using $\eta(x) \leq \eta(x+1) - 1$ it follows that the quotient can never attain the value 1, for $x < \bar{x}$.

Finally, since S is a minimal cover and $d^q \in (0,1]$ for all $q \in S$, it follows that $\bar{x} = D(Q^1 \cup S) - 1$. Substitution of this value in the inequalities as described in Theorem 6.6.1 and applying the Shifting Theorem 6.4.2 yields the required inequalities. ∎

Example 6.6.1 *Consider an instance of the edge capacity polytope with three items, and let $d^1 = 0.4, d^2 = 0.4, d^3 = 1.4$. First we transform the data such that $d^q \in (0,1]$ for all $q \in Q$ hence we consider the data $r^1 = r^2 = r^3 = 0.4$. Let $Q^0 = Q^1 = \emptyset, S = \{1,2,3\}$ and define $\bar{x} = 1$. Hence, we are analyzing the set $X(Q^0, Q^1, \bar{x})$ with the knapsack inequality*

$$0.4f^1 + 0.4f^2 + 0.4f^3 \leq 1$$

for which S is a minimal cover. Applying integer lifting to the cover inequality $f^1 + f^2 + f^3 \leq 2$ leads to lifting coefficients $\alpha^L = 1, \alpha^U = 2$, and the resulting inequalities read

$$\begin{aligned} x &\geq f^1 + f^2 + f^3 - 1 \\ 2x &\geq f^1 + f^2 + f^3 \end{aligned}$$

which are facet defining for the edge capacity polytope on the transformed data r^1, r^2, r^3. Applying the shifting theorem to obtain inequalities which are facet defining for X with the original data leads to the facet defining inequalities

$$\begin{aligned} x &\geq f^1 + f^2 + 2f^3 - 1 \\ 2x &\geq f^1 + f^2 + 3f^3. \end{aligned}$$

Note that if we had applied the same technique to the original demand data directly, and for the value $\bar{x} = 2$, again the set S is a minimal cover for the associated knapsack polytope. However, if integer lifting is applied to the cover inequality $f^1 + f^2 + f^3 \leq 2$ in this case, then the lifting coefficients are $\alpha^L = 1, \alpha^U = 0$, implying that integer lifting would not be possible.

6.7 C-strong Inequalities

This section analyzes lifted cover inequalities for the value $\alpha^L = 1$. We will show that the lifting of fixed 0–1 variables in the sets Q^0 and Q^1 can be done efficiently and that the resulting inequalities are equivalent to c-strong inequalities as described by Brockmüller, Günlück and Wolsey [20]. Subsection 6.7.1 states some new properties of c-strong inequalities, which indicate the importance of this class of valid inequalities in the polyhedral description of the edge capacity polytope.

Theorem 6.7.1 *Let Q^0, Q^1, S, T be a partition of the set Q. If*

$$x \geq \sum_{q \in S} D^q f^q + \sum_{q \in T} (D^q - 1) f^q - c$$

is a facet defining inequality for $conv(X(Q^0, Q^1))$ then

(i). $c = n(Q^1, S) - D(Q^1)$

(ii). if maximal lifting is applied to $\hat{q} \in Q^0$ to obtain a facet defining inequality

$$x \geq \textstyle\sum_{q \in S} D^q f^q + \sum_{q \in T} (D^q - 1) f^q + \beta^{\hat{q}} f^{\hat{q}} - c$$

for $conv(X(Q^0 \setminus \{\hat{q}\}, Q^1))$ *then*

$$\beta^{\hat{q}} = \begin{cases} D^{\hat{q}} & \text{if } r^{\hat{q}} > 1 - r(S \cup Q^1) \\ D^{\hat{q}} - 1 & \text{otherwise} \end{cases}$$

(iii). if maximal lifting is applied to $\hat{q} \in Q^1$ *to obtain a facet defining inequality*

$$x \geq \sum_{q \in S} D^q f^q + \sum_{q \in T} (D^q - 1) f^q + \beta^{\hat{q}} (f^{\hat{q}} - 1) - c$$

for $conv(X(Q^0, Q^1 \setminus \{\hat{q}\}))$ *then*

$$\beta^{\hat{q}} = \begin{cases} D^{\hat{q}} & \text{if } r^{\hat{q}} > 1 - r(S \cup Q^1 \setminus \{\hat{q}\}) \\ D^{\hat{q}} - 1 & \text{otherwise} \end{cases}$$

(iv). the lifting can be done in polynomial time and leads the facet defining inequality for $conv(X)$*, for a certain partition* $\bar{Q}, Q \setminus \bar{Q}$ *of the set* Q*:*

$$x \geq \sum_{q \in \bar{Q}} D^q f^q + \sum_{q \in Q \setminus \bar{Q}} (D^q - 1) f^q - n(\emptyset, \bar{Q})$$

Proof. If the inequality is facet defining then it must be tight for some solutions. Hence,

$$\begin{aligned} c &= \max_{(x,f) \in X(Q^0, Q^1)} \left\{ \sum_{q \in S} D^q f^q + \sum_{q \in T} (D^q - 1) f^q - x \right\} \\ &= \sum_{q \in S} D^q - D(Q^1 \cup S) = n(Q^1, S) - D(Q^1) \end{aligned}$$

Next, if $\hat{q} \in Q^0$ is lifted, then the lifting problem reads

$$\beta^{\hat{q}} = \min_{(x,f) \in X(Q^0 \setminus \{\hat{q}\}, Q^1 \cup \{\hat{q}\})} \left\{ x - \sum_{q \in S} D^q f^q - \sum_{q \in T} (D^q - 1) f^q + c \right\}$$

and the minimum for this problem is attained for $f^q = 1$ if $q \in S$ and $f^q = 0$ if $q \in T$, and $\bar{x} = D(Q^1 \cup S \cup \{\hat{q}\})$. This implies that

$$\begin{aligned} \beta^{\hat{q}} &= D(Q^1 \cup S \cup \{\hat{q}\}) - \sum_{q \in S} D^q + n(Q^1, S) - D(Q^1) \\ &= D(Q^1 \cup S \cup \{\hat{q}\}) - D(Q^1 \cup S) \end{aligned}$$

which yields the required result. If $\hat{q} \in Q^1$ is lifted, a similar reasoning holds. The fact that lifting can be done in polynomial time now follows directly. ■

These lower convex envelope inequalities with slope 1 are the so-called c-strong inequalities as developed by Brockmüller, Günlück and Wolsey [20], which we will redefine in the next subsection.

6.7.1 Properties of C-strong Inequalities

In this section we list some properties of c-strong inequalities which indicate the importance of this class of valid inequalities in order to get a good approximation of $conv(X)$. Before we do so, we repeat some definitions and results from Brockmüller, Günlük and Wolsey [20].

Definition 6.7.1 *(cf. [20]) A set $S \subseteq Q$ is called c-strong if $c = \sum_{q\in S} D^q - D(S)$. The set is maximal c-strong if $S \setminus \{i\}$ is c-strong, for all $i \in S$, and $S \cup \{i\}$ is not c-strong for all $i \in Q \setminus S$. Note that a set S is c-strong if and only if $c \leq \sum_{q\in S}(1 - r^q) < (c + 1)$. Moreover, $c = n(\emptyset, S)$.*

Proposition 6.7.1 *(cf. [20]) Let $S \subseteq Q$ be a c-strong set. Then*

$$x \geq \sum_{q\in S} D^q f^q + \sum_{q\in Q\setminus S} (D^q - 1)f^q - c \tag{6.28}$$

is a facet defining inequality for $conv(X)$ if and only if S is maximal c-strong.

Lemma 6.7.1 *Let $S \subseteq Q$ be a c-strong subset, with $S \cup \{i\}$ $(c+1)$-strong for all $i \in Q \setminus S$. If $S \setminus \{\bar{q}\}$ is $(c - 1)$ strong for $\bar{q} \in S$, then $(S \setminus \{\bar{q}\}) \cup \{i\}$ is c-strong, for all $i \in Q \setminus S$.*

Proof. Let $i \in Q \setminus S$. The fact that $S \cup \{i\}$ is $(c + 1)$-strong implies that $(c + 1) \leq \sum_{q\in S\cup\{i\}}(1 - r^q) < (c + 2)$. Similarly, since $S \setminus \{\bar{q}\}$ is $(c - 1)$-strong implies that $(c - 1) \leq \sum_{q\in S\setminus\{\bar{q}\}}(1 - r^q) < c$. Suppose that $\sum_{q\in(S\setminus\{\bar{q}\})\cup\{i\}}(1 - r^q) < c$, then a contradiction is obtained since $(1 - r^{\bar{q}}) \in [0, 1)$ and $\sum_{q\in S\cup\{i\}}(1 - r^q) \geq (c+1)$. Suppose that $\sum_{q\in(S\setminus\{\bar{q}\})\cup\{i\}}(1 - r^q) \geq (c + 1)$, then a contradiction is obtained since $(1 - r^{\bar{q}}) \in [0, 1)$ and $\sum_{q\in S\setminus\{\bar{q}\}}(1 - r^q) < c$. Hence, $c \leq \sum_{q\in(S\setminus\{\bar{q}\})\cup\{i\}}(1 - r^q) < (c + 1)$, which proves our claim. ■

Proposition 6.7.2 *Let $x \geq \sum_{q\in Q} b^q f^q - c$, $(c \in \mathbb{N}, b^q \in \mathbb{N}$ for all $q \in Q)$ be a facet defining inequality for conv(X). Then this inequality is a c-strong inequality.*

Proof. Follows directly from Theorem 6.7.1 and Theorem 6.4.3. ■

Proposition 6.7.3 *Each vertex of $conv(X)$ is on a face defined by a facet defining c-strong inequality.*

Proof. Let $(\bar{x}, \bar{f})$ be a vertex of $conv(X)$, and let $\bar{S} := \{q \in Q : \bar{f}^q = 1\}$. Let $c = \sum_{q\in\bar{S}} D^q - D(\bar{S})$, such that $\bar{S}$ is c-strong. Then $(\bar{x}, \bar{f})$ is on the face defined by the inequality $x \geq \sum_{q\in\bar{S}} D^q f^q + \sum_{q\in Q\setminus\bar{S}}(D^q - 1)f^q - c$. If there exists a $\hat{q} \in Q \setminus S$ such that

$\bar{S} \cup \{\hat{q}\}$ is c-strong, add $\hat{q}$ to the set $\bar{S}$. Repeat this process until no items can be added to the set $\bar{S}$ without violating the fact that the set remains c-strong. Let S_1 represent this new set of commodities. Note that $(\bar{x}, \bar{f})$ is also on the face defined by the inequality $x \geq \sum_{q \in S_1} D^q f^q + \sum_{q \in Q \setminus S_1} (D^q - 1) f^q - c$. Next, if there exists a $\tilde{q} \in S_1$ such that $S_1 \setminus \{\tilde{q}\}$ is not c-strong but $(c-1)$-strong, then remove $\tilde{q}$ from S_1. After this removal, $(\bar{x}, \bar{f})$ is on the face defined by the inequality $x \geq \sum_{q \in S_1 \setminus \{\tilde{q}\}} D^q f^q + \sum_{q \in (Q \setminus S_1) \cup \{\tilde{q}\}} (D^q - 1) f^q - (c-1)$. Again, repeat this process until one obtains a set S_2 which is c_2-strong and such that $S_2 \setminus \{q\}$ is c_2-strong, for all $q \in S_2$. Lemma 6.7.1 implies that $S_2 \cup \{q\}$ is more than c_2-strong, such that S_2 is indeed maximal c_2-strong. Hence the inequality $x \geq \sum_{q \in S_2} D^q f^q + \sum_{q \in Q \setminus S_2} (D^q - 1) f^q - c_2$ defines a facet of $conv(X)$. ■

Moreover, in several special cases each facet of the edge capacity polytope is either a model inequality or a c-strong inequality.

Proposition 6.7.4 *If the set Q is 0-strong, then $conv(X)$ is completely described by the trivial inequalities and the 0-strong inequality $x \geq \sum_{q \in Q} D^q f^q$.*

Proof. Given an arbitrary objective function $(\delta, \gamma) \in \mathbb{Z} \times \mathbb{Z}^{|Q|}$ which is to be minimized over all solutions in X, let $M(\delta, \gamma)$ be the corresponding set of optimal solutions. We will show that for each possible vector (δ, γ), the set $M(\delta, \gamma)$ is a subset of a face described by either one of the trivial inequalities or the 0-strong inequality. We distinguish a number of cases.

- $\delta < 0$, then the primal solution is unbounded, hence $M(\delta, \gamma) = \emptyset$.
- $\delta \geq 0, \gamma^{\hat{q}} > 0$, for some $\hat{q} \in Q$. Then $M(\delta, \gamma) \subseteq \{(x, f) : f^{\hat{q}} = 0\}$.
- $\delta = 0, \gamma^{\hat{q}} < 0$, for some $\hat{q} \in Q$, then $M(\delta, \gamma) \subseteq \{(x, f) : f^{\hat{q}} = 1\}$.
- $\delta = 0, \gamma^q = 0$, for all $q \in Q$, then $M(\delta, \gamma)$ equals the set X itself.
- $\delta > 0, \gamma^q = 0$, for all $q \in Q$. Then $M(\delta, \gamma) \subseteq \{(x, f) : f^q = 0\}$ for all $q \in Q$.
- $\delta > 0, \gamma^q = 0$ for all $q \in T$, $\gamma^q < 0$ for all $q \in Q \setminus T$. Let $(\bar{x}, \bar{f}) \in M(\delta, \gamma)$ be an arbitrary optimal solution, and let $\bar{S} := \{q \in Q : \bar{f}^q = 1\}$. Since Q is 0-strong, the same holds for the set $\bar{S}$. Moreover, since $\delta > 0$, $\bar{x} = D(\bar{S}) = \sum_{q \in \bar{S}} D^q$. Hence, the solution $(\bar{x}, \bar{f})$ satisfies the 0-strong inequality at equality.

■

Proposition 6.7.5 *If the set Q is $|Q| - 1$-strong, then $conv(X)$ is completely described by the trivial inequalities and the 0-strong inequalities $x \geq D^q f^q$, for all $q \in Q$.*

Proof. Given an arbitrary objective function $(\delta, \gamma) \in \mathbb{R} \times \mathbb{R}^{|Q|}$ which is to be minimized over all solutions in X, let $M(\delta, \gamma)$ be the corresponding set of optimal solutions. We will

show that for each possible vector (δ, γ), the set $M(\delta, \gamma)$ is a subset of a face described by either one of the trivial inequalities or one of the 0-strong inequalities. We distinguish the same cases as in the proof of Proposition 6.7.4. The first 5 cases are analogous, so we restrict ourselves to the last case. Hence, $\delta > 0$, $\gamma^q = 0$, for all $q \in T$ and $\gamma^q < 0$ for all $q \in Q \setminus T$. Let $(\bar{x}, \bar{f})$ be an arbitrary optimal solution in $M(\delta, \gamma)$. Note that $\bar{x} \leq 1$, since the set is $|Q| - 1$-strong and $\delta > 0$. Let $\hat{q} = \arg\min_{q \in Q} \gamma^q$ (hence $\gamma^{\hat{q}} < 0$). If $\bar{x} = 1$ holds in an optimal solution, then surely $\bar{f}^{\hat{q}} = 1$, and if $\bar{x} = 0$ in an optimal solution, then $\bar{f}^{\hat{q}} = 0$. Hence, $M(\delta, \gamma) \subseteq \{(x, f) : x \geq D^{\hat{q}} f^{\hat{q}}\}$. ■

Corollary 6.7.1 *If $|Q| \leq 2$ then the polytope $conv(X)$ is completely described by the trivial inequalities and c-strong inequalities.*

Proof. If $|Q| \leq 2$, then the set Q is either 0-strong or $|Q| - 1$-strong, hence, the previous propositions prove our claim. ■

6.8 The Directed Edge Capacity Polytope

This section reports on the directed edge capacity polytopes Y_{ij}^{DF} and Y_{ij}^{DP}, as described in Section 6.2. We derive a class of valid inequalities and identify conditions under which the valid inequalities are facet defining. Next, we show that sequential lifting for the directed edge capacity polytope is NP-hard in general, even for lifting orders which first lift all flow variables in the same direction.

Define the directed edge capacity polytope Y, where f^q is a binary variable that denotes whether commodity q is routed on the forward arc and h^q represents whether the commodity q is routed on the backward arc, as follows:

$$Y = \{(x, f, h) \in \mathbb{N} \times \{0,1\}^{2|Q|} : x \geq \textstyle\sum_{q \in Q} d^q f^q, x \geq \sum_{q \in Q} d^q h^q\}$$

Proposition 6.8.1 *Let $\hat{q} \in Q$ and let $\alpha \in \mathbb{N}$ such that $1 \leq \alpha \leq D^{\hat{q}}$. Then*

$$x \geq \alpha f^{\hat{q}} + \textstyle\sum_{q \in Q} (D^q - \alpha) h^q \tag{6.29}$$

is a valid inequality for Y.

Proof. Consider an arbitrary feasible solution $(\bar{x}, \bar{f}, \bar{h}) \in Y$ and let $\bar{Q} = \{q \in Q : \bar{h}^q = 1\}$. If $|\bar{Q}| = 0$, then $\bar{x} \geq D^{\hat{q}} \bar{f}^{\hat{q}} \geq \alpha \bar{f}^{\hat{q}} = \alpha \bar{f}^{\hat{q}} + \sum_{q \in Q} (D^q - \alpha) \bar{h}^q$. If $|\bar{Q}| \geq 1$, then

$$\begin{aligned}
\bar{x} &\geq \lceil \max\{\textstyle\sum_{q \in Q} d^q \bar{f}^q, \sum_{q \in Q} d^q \bar{h}^q\} \rceil \geq D(\bar{Q}) \\
&= \textstyle\sum_{q \in \bar{Q}} (D^q - \alpha) + \sum_{q \in \bar{Q}} (\alpha - 1) + \lceil \sum_{q \in \bar{Q}} r^q \rceil \\
&\geq \textstyle\sum_{q \in \bar{Q}} (D^q - \alpha) + (\alpha - 1) + 1 \geq \alpha \bar{f}^{\hat{q}} + \sum_{q \in Q} (D^q - \alpha) \bar{h}^q
\end{aligned}$$

which proves our claim. ■

Proposition 6.8.2 *Let $\hat{q} \in Q$. If*

(i). $\alpha = 1$

(ii). $\forall q \in Q : D^q \geq D^{\hat{q}}$

(iii). $\forall q \in Q \setminus \{\hat{q}\} : \exists \tilde{Q} \subseteq Q$, *with* $|\tilde{Q}| = 2$, $\left\lceil \sum_{q \in \tilde{Q}} r^q \right\rceil = 1$, *and* $D(\tilde{Q}) \geq D(\{q, \hat{q}\})$

then (6.29) is facet defining for $conv(Y)$.

Proof. The dimension of $conv(Y)$ is $2|Q| + 1$. We give $2|Q|$ affinely independent vectors in Y satisfying the inequality at equality. These vectors are

- $(x, f, h) = (0, 0, 0)$
- $(x, f, h) = (D^q, e^{\hat{q}}, e^q)$, for all $q \in Q$
- $(x, f, h) = (D(\tilde{Q}), e^{\hat{q} \cup q}, e^{\tilde{Q}})$, for all $q \in Q \setminus \{\hat{q}\}$

■

Example 6.8.1 *Consider an instance of the set Y with $|Q| = 4$ and the demands are $d^1 = 1.8, d^2 = 2.1, d^3 = 2.6, d^4 = 3.2$. If $\hat{q} = 1$, then the inequalities*

$$x \geq f^1 + h^1 + 2h^2 + 2h^3 + 3h^4$$
$$x \geq h^1 + f^1 + 2f^2 + 2f^3 + 3f^4$$

are valid and facet defining for $conv(Y)$.

As follows from Section 6.7, 0-strong inequalities which are facet defining for $conv(X)$ can be obtained by starting with the valid inequality $x \geq 0$ for $conv(X(Q, \emptyset))$ (i.e. the polytope in which all f variables are set equal to zero), and then applying sequential lifting to the set of commodities Q. This lifting can be done in polynomial time as a result of Theorem 6.7.1. A similar approach could be employed to obtain facet defining inequalities for the directed polytope $conv(Y)$. First one could fix all variables (both f and h variables) to zero, and next one could apply sequential lifting on the valid inequality $x \geq 0$. However, we show that lifting under an arbitrary lifting order is an NP-hard problem in general.

SUBSET SUM (cf. Garey and Johnson [37])
INSTANCE: A set of items A, a size $s^q \in \mathbb{N}$ for all $q \in A$, and a positive integer B.
QUESTION: Does there exist a subset $\bar{A} \subseteq A$ such that $\sum_{q\in\bar{A}} s^q = B$?

LIFTING FOR DIRECTED EDGE CAPACITY MODEL
INSTANCE: A set of commodities Q, a demand size $d^q \in \mathbb{N}$ for all $q \in Q$, and a capacity $\lambda \in \mathbb{N}$ (this defines an instance of Y, using the inequalities $x \geq \sum_{q\in Q} \tilde{d}^q f^q$ and $x \geq \sum_{q\in Q} \tilde{d}^q h^q$, where $\tilde{d}^q = d^q/\lambda$). A complete order π on a set of variables $T = \cup_{q\in Q}(f^q \cup h^q)$, a specific variable $z \in T$ and an integer $K \in \mathbb{N}$.
QUESTION: If maximal lifting is applied to the inequality $x \geq 0$ in the lifting order π to obtain a facet defining inequality for Y, is the lifting coefficient of variable z less than or equal to K?

Proposition 6.8.3 LIFTING FOR DIRECTED EDGE CAPACITY MODEL *is NP-complete.*

Proof. We show that SUBSET SUM polynomially reduces to LIFTING FOR DIRECTED EDGE CAPACITY MODEL. Given an instance of SUBSET SUM, construct an instance of LIFTING FOR DIRECTED EDGE CAPACITY MODEL as follows. Let $Q = \{1,2,3\} \cup A$, $\lambda = B$, and define $d^q = s^q(\lambda+1)$, for all $q \in A$. Define an integer $m = \left\lceil \frac{\sum_{q\in A} d^q}{\lambda} \right\rceil$, and let $d^1 = (m+1)\lambda + 1, d^2 = (m+1)\lambda + \lambda - 1, d^3 = (2m+4)\lambda + \lambda^2$. Define the order π on T as $f^1, f^2, f^q, \forall q \in A, h^3, h^2, h^1, h^q, \forall q \in A, f^3$. Finally, let $z = h^3$ and $K = 1$.

Next we show that an instance for SUBSET SUM yields an affirmative answer if and only if the corresponding instance for LIFTING FOR DIRECTED EDGE CAPACITY MODEL yields an affirmative answer. For convenience, we will use $\tilde{d}^q = d^q/\lambda$ throughout the proof. First, note that, starting with $x \geq 0$, after lifting the variables $f^1, f^2, f^q, \forall q \in A$ the valid inequality reads

$$x \geq b^T f = \lceil \tilde{d}^1 \rceil f^1 + (\lceil \tilde{d}^2 \rceil - 1) f^2 + \sum_{q\in A} (\lceil \tilde{d}^q \rceil - 1) f^q$$

Next, if we apply maximal lifting to variable h^3 and if $\bar{A} \subseteq A$ is a subset with $\sum_{q\in\bar{A}} s^q = B$, then the lifting coefficient b^3 for the variable h^3 satisfies

$$\begin{aligned} b^3 &= \min_{(x,f,h)\in Y : h^3=1, f^3=0, h^q=0, \forall q\in Q\setminus\{3\}} \{x - (b^T f)\} \\ &\leq \left\lceil \max\{\tilde{d}^{\hat{q}}, \tilde{d}(\{1,2\} \cup \bar{A})\} \right\rceil - \sum_{q\in\{1,2\}\cup\bar{A}} b^q \\ &= 2m + 4 + \lambda - (2m + 3 + \lambda) = 1 = K \end{aligned}$$

Conversely, let $(\bar{x}, \bar{f}, \bar{h})$ be the vector for which the minimum value less than or equal to $K = 1$ in the lifting problem is attained, and let $\bar{Q} = \{q \in Q : \bar{f}^q = 1\}$. If $1 \notin \bar{Q}$, then $\sum_{q\in\bar{Q}} \tilde{d}^q \leq \tilde{d}^{\hat{q}}$ and hence,

$$\begin{aligned} b^3 &= \min_{(x,f,h)\in Y : h^3=1, f^3=0, h^q=0, \forall q\in Q\setminus\{3\}} \{x - (b^T f)\} \\ &\geq \tilde{d}^{\hat{q}} - (b^T f) = 2m + 4 + \lambda - (m + 1 + \sum_{q\in A} s^q) \geq 3 + \lambda > K \end{aligned}$$

since $m \geq \sum_{q \in A} s^q$. Hence, $\bar{f}^1 = 1$. Similarly, one can prove that $\bar{f}^2 = 1$. Next define $\tilde{A} = \{q \in A : \bar{f}^q = 1\}$, and let $p = \sum_{q \in \tilde{A}} s^q$. If $p < B$, then $\tilde{d}(\{1,2\} \cup \tilde{A}) < \tilde{d}^{\tilde{q}}$, thus,

$$b^3 = \bar{x} - \sum_{q \in \tilde{Q}} b^q = 2m + 4 + \lambda - (2m + 3 + p) = \lambda - p + 1 > K$$

so, $p < B$ cannot be the case. If $p > B$, then $\tilde{d}(\{1,2\} \cup \tilde{A}) > \tilde{d}^{\tilde{q}}$, hence,

$$b^3 = \bar{x} - \sum_{q \in \tilde{Q}} b^q = 2m + 3 + p + \left\lceil \frac{p}{\lambda} \right\rceil - (2m + 3 + p) = \left\lceil \frac{p}{\lambda} \right\rceil \geq 2 > K$$

hence, neither $p > B$ can be the case. But this yields that $\sum_{q \in \tilde{A}} s^q = p = B$, hence $\tilde{A}$ is the required subset. ■

6.9 Computational Issues

To test the effect of the developed theory on the solvability of network loading problems we implemented a Branch-and-Cut algorithm, using the Branch-and-Cut system ABACUS, version 1.2 [74], in combination with CPLEX 4.0 [48]. The algorithm was executed on a DEC 2100 A500MP workstation with 128MB internal memory, and tested on a set of instances for the *DNFM* (or *DNPM*) model made available to us by KPN Research, Leidschendam, The Netherlands. These instances are defined on complete graphs in the range of 4 to 8 nodes, and for each graph size three different instances with fully dense non-symmetric demand matrices were available.

The flow formulation *DNFM* as described by (6.9)-(6.13) and the path formulation *DNPM* as described by (6.14)-(6.18) both model the same directed network loading problem. For a suitable choice of the set of paths for the commodities, and the exclusion of cycles from the flow formulation, it holds that every feasible solution to the former formulation corresponds to a feasible solution to the latter formulation, and vice versa. Consequently, the optimal values are the same. More importantly, this property also holds for the LP-relaxations of the formulations, even after the addition of valid inequalities as discussed in the sequel. Still, we implemented both formulations to test whether either of the two formulations would yield better results due to the difference in number of variables. Moreover, standard branching strategies use fixing of variables. If a path variable related to a certain commodity is fixed to its upper bound, then the routing of that specific commodity is completely known. Fixing a flow variable for a certain commodity to its upper bound only gives limited information on the routing for that specific commodity. Hence, this might also lead to different running times for both formulations. Although it is hard to draw general conclusions from the limited set of instances available to us, our computational results indicated that for larger graphs the exponential growth of the number of path variables is a serious problem. The computational results stated in the sequel are therefore obtained using the flow formulation.

6.9.1 Separation of Valid inequalities

Brockmüller, Günlük and Wolsey [20] describe the separation problem for c-strong inequalities, and show that for a given value of c, finding the most violated c-strong inequality requires solving a knapsack problem. They propose heuristic methods to find the most violated inequality for values $c = 0, 1, 2$, and our computational experiments support their findings that this method yields good results.

Although lifting of the more general class of lower convex envelope inequalities can be performed in polynomial time (as described in Section 6.5), there remain several unsolved issues regarding the separation of these inequalities. In order to obtain the most violated inequality in this class, it is yet unclear what choice of starting polytope $X(Q^0, Q^1, \bar{x})$ should be employed, which lifting order to use, and, given the potential danger of high computation times for exact lifting, whether lifting should be done exactly or by heuristics. We therefore propose to search for violated lower convex envelop inequalities as follows. For a given LP-solution $(\bar{x}, \bar{f})$ and an arbitrary arc $(i, j) \in A$, let S be the set of commodities with positive (possibly fractional) routing variables on the arc ($S = \{q \in Q : \bar{f}^q_{ij} > 0\}$). Then one can easily construct a two-dimensional picture as in Figure 6.1 containing all feasible solutions on the subset of variables $(\cup_{q \in S} f^q_{ij}) \cup x$, and define all corresponding lower convex envelop inequalities for this subset of variables. Our computational experiments indicate that constructing these lower convex envelop inequalities and checking whether any of these inequalities are violated can be done efficiently.

Next, for a given arc $(i, j) \in A$ and a fixed value of α, finding the most violated two-side inequality (6.29) is an easy task. For a given LP-solution $(\bar{x}, \bar{f})$ a violated two-side inequality exists if and only if there exists an element $\hat{q} \in Q$ such that $\alpha \bar{f}^{\hat{q}}_{ij} > \bar{x} - \sum_{q \in Q} (D^q - \alpha) \bar{h}^q_{ji}$. Since the right hand side of the latter inequality is a constant for the given LP-solution, finding the most violated two-side inequality on arc (i, j) for the specific value of α (if one exists) is equivalent to finding the maximal value $\bar{f}^q_{ij}$ over all commodities $q \in Q$. This can be done by any sorting algorithm. Likewise as for c-strong inequalities, computational experiments indicate that two-side inequalities should only be considered for small values of α, for instance $\alpha \in \{1, 2\}$.

Apart from the inequalities for the edge capacity polytope as described in this chapter, we also incorporated some other classes of well-known valid inequalities for network loading problems. Cut-set inequalities are used quite extensively for network loading problems (see for instance Barahona [12], Magnanti, Mirchandani and Vachani [55], [56], Bienstock and Günlük [16], among others). Given a partition of the node set V into two sets S and T, let $d[S, T]$ denote the accumulated demand of all commodities with source node in S and sink node in T. Then it is clear that the total capacity on the edges in the cut $\delta[S, T]$ should exceed this accumulated demand since all of these commodities must cross the cut. Since, capacity can only be installed in integer amounts, the cut-set inequalities read

$$\sum_{\{i,j\} \in \delta[S,T]} x_{ij} \geq \max\{\lceil d[S,T] \rceil, \lceil d[T,S] \rceil\} \tag{6.30}$$

Likewise, three partition inequalities (based on a partition of the node set into three sets)

have been considered (see [16]), as well as the general K-cuts (see Barahona [13]). For small to medium sized graphs as considered in our experiments an exact separation that considers all possible partitions of the graph can be performed reasonably fast, and this strategy is used in our computations.

6.9.2 Computational Results and Future Research

Table 6.1 shows the results of the Branch-and-Cut algorithm for the fifteen real-life instances from KPN Research. The name of each instance, stated in the first column, refers to the number of nodes in the graph (first digit), whereas the second number in the name defines the demand matrix. The input for the algorithm included a lower bound for the problem instance obtained with the heuristics of Chapter 5, stated in column 3 of the table. The time associated with these heuristics is not included in the time measurements indicated in the table. Moreover, simple rounding techniques were used in the nodes of the search tree to generate integer solutions from the fractional LP-solutions and hence, additional upper bounds were obtained during the process. Column 2 represents the best integer value found by the algorithm. Moreover, the table states the LP-value in the root node, the LP-value in the root node after the addition of valid inequalities, and the reduction of the initial gap between LP and IP value. Finally, the total CPU time, the number of added valid inequalities and the size of the Branch-and-Cut tree are listed in Table 6.1. Note that for the instances defined on the complete graph with 8 nodes, the algorithm was not able to prove optimality of the listed solutions, due to memory restrictions. These results indicate that, although the added valid inequalities yield a large reduction in the gap between LP and IP-value in the root node, more research is required to obtain additional knowledge on the polyhedral structure of non-bifurcated network loading problems.

Given these computational results, the question arises how much gain can be obtained for the solution process of network loading problems from additional research (and insight) on the polyhedral structure of the edge capacity polytope itself. To answer this question, one would like to compare the total set of violated facet defining inequalities for the edge capacity polytope for a given LP-solution with the set of violated inequalities as obtained by the separation algorithms and added during the Branch and Cut algorithm. Once the difference is known, one could measure the impact of these additional inequalities on the Branch-and-Cut algorithm. Given the definition of an edge capacity polytope associated with a specific edge, there exist well-known algorithms which, given the extreme points of the polytope, yield the complete polyhedral description of the polytope (see for instance Fukuda [34]). In general, these algorithms are only computationally feasible for polytopes with a limited dimension or number of extreme points, and therefore such algorithms are often used to obtain insight into polyhedral structures for small polytopes (however, see Christof and Reinelt [24] for an enumeration approach that incorporates such techniques).

For most LP-solutions encountered during the Branch-and-Cut process the number of commodities with positive flow on an arc or edge is relatively small. Moreover, the degree of violation of any facet defining inequality for the edge capacity polytope is only affected by variables with positive (fractional) value. Hence, in order to test whether there exist

Table 6.1: Computational results for KPN network loading instances.

Instance	Best IP	Heur	LP	LP+vi	gap%	#sec	#vi	#BC
I_4_3	**3**	3	1.74	3.000	100%	0.15	34	1
I_4_10	**8**	8	5.81	7.125	60%	0.51	93	7
I_4_20	**13**	13	11.6	13.00	100%	0.20	42	1
I_5_3	**4**	4	1.89	3.750	88%	0.83	123	3
I_5_10	**9**	9	6.32	8.135	68%	1.68	181	7
I_5_20	**16**	16	12.7	15.01	70%	3.88	283	15
I_6_3	**5**	5	1.93	4.109	71%	7.16	437	11
I_6_10	**9**	10	6.45	8.377	76%	33.0	1273	41
I_6_20	**16**	17	12.9	15.46	83%	92.8	3554	127
I_7_3	**6**	6	1.95	4.417	81%	76.7	1833	25
I_7_10	**10**	11	6.52	8.850	67%	1005	20422	453
I_7_20	**17**	18	13.0	15.94	74%	415	7093	255
I_8_3	7	7	2.17	4.880	56%	XX		
I_8_10	12	12	7.23	9.890	56%	XX		
I_8_20	21	21	14.5	17.56	47%	XX		

violated facet defining inequalities for the edge capacity polytope for a given LP-solution, one only needs to test whether there exist violated facet defining inequalities for the edge capacity polytope on this restricted set of commodities. Therefore, we performed the following computational experiment. If at any node of the Branch and Cut tree no more violated cuts could be found, we generated the extreme points of the edge capacity polytope defined on the restricted set of commodities with positive flow on the arc (or edge), and used Fukuda's software package CDD to obtain the complete polyhedral description of this restricted edge capacity polytope. Next, we enumerate this set of facet defining inequalities to check whether there are any violated inequalities that can be added to the formulation of the network loading problem. Usually, the number of violated inequalities found by this procedure is very small (typically in the range 0–5) and the effect of adding these inequalities to the formulation is marginal. Hence, the potential gain of such inequalities is expected to be small.

Future research on non-bifurcated network loading problems could therefore focus on valid inequalities for more general structures of the integer programming formulation, such as cuts in the graph. Although cut-set inequalities (or the more general partition inequalities) are facet defining for bifurcated versions of network loading problems (see [56],[16]), for non-bifurcated network loading problems this is in general not the case, and several possibilities for strengthening arise. We have performed some initial computational experiments to test the effect of a strengthening of a cut-set inequality, but so far the gain has been limited.

Next, a special observation can be made regarding the IP-formulation of non-bifurcated network loading problems. These formulations contain two types of constraints (routing

and capacity constraints), as well as two types of variables (routing and capacity variables). The non-bifurcated network loading problem formulation can therefore be viewed as follows. The feasibility of the routing variables is determined only by the routing constraints, and given a feasible choice of the routing variables, the capacity constraints merely define the capacity variables, and therefore the objective value. Hence, there is no interaction between any of the capacity variables in the formulation. As a consequence, one can prove that a facet defining inequality for the network loading problem formulation with only a subset of the capacity constraints yields a facet defining inequality for the overall problem (using the same techniques as in Section 6.3). Due to the limited interaction between the capacity variables and capacity constraints, it may well be that an important gain in the improvement of polyhedral descriptions of the network loading polytope should be achieved by general purpose cuts, such as Chvatal-Gomory cuts (see Marchand and Wolsey [59] for general purpose polyhedral techniques in solving mixed integer programs).

Bibliography

[1] E.H.L. Aarts and J.K. Lenstra. *Local Search in Combinatorial Optimization*. Wiley, N.Y., 1997.

[2] W. Ackermann. "Zum hilberschen aufbau der reellen zahlen". *Mathematische Annalen*, 99:118–133, 1928.

[3] E.H. Aghezzaf, T.L. Magnanti, and L.A. Wolsey. "Optimizing constrained subtrees of trees". *Mathematical Programming*, 71:113–126, 1995.

[4] D. Alevras, M. Grötschel, and R. Wessäly. "Capacity and survivability models for telecommunication networks". Preprint SC 97-24, Konrad-Zuse-Zentrum fur Informationstechnik Berlin, June 1997.

[5] A. Amiri and H. Pirkul. "Primary and secondary route selection in backbone communication networks". *European Journal of Operational Research*, 93:98–109, 1996.

[6] A. Amiri and H. Pirkul. "Routing and capacity assignment in backbone communication networks". *Computers and Operational Research*, 24:275–287, 1997.

[7] A. Balakrishnan, T. Magnanti, J. Sokol, and Y. Wang. "Modelling and solving the single facility line restoration problem". Working paper, 1997.

[8] A. Balakrishnan, T.L. Magnanti, and R.T. Wong. Models for planning capacity expansion in local access telecommunication networks. *Annals of Operations reserch*, 33:239–284, 1991.

[9] A. Balakrishnan, T.L. Magnanti, and R.T. Wong. "A decomposition algorithm for local access telecommunications network expansion planning". *Operations Research*, 43(1):58–76, 1995.

[10] E. Balas. "Facets of the knapsack polytope". *Mathematical Programming*, 8:146–164, 1975.

[11] E. Balas and E. Zemel. "Facets of the knapsack polytope from minimal covers". *SIAM Journal of Applied Mathematics*, 34:119–148, 1978.

[12] F. Barahona. "Network design using cut inequalities". *SIAM journal on Optimization*, 6:823–837, 1996.

[13] F. Barahona. "On the k-cut problem". Working paper, 1998.

[14] I. Barany, J. Edmonds, and L.A. Wolsey. "Pacing and covering a tree by subtrees". *Combinatorica*, 6:221–233, 1986.

[15] D. Bienstock, S. Chopra, O. Günlük, and C.-Y. Tsai. "Minimum cost capacity installation for multicommodity network flows". Working paper, July 1995.

[16] D. Bienstock and O. Günlük. "Capacitated network design — polyhedral structure and computation". Working paper, June 1995.

[17] D. Bienstock and G. Muratore. "Strong inequalities for capacitated survivable network design problems". Working Paper, December 1997.

[18] D. Bienstock and I. Saniee. "Atm network design: Traffic models and optimization based heuristics". Working Paper, 1997.

[19] E.A. Boyd. "Polyhedral results for the precedence constrained knapsack problem". *Discrete Applied Mathematics*, 41:185–201, 1993.

[20] B. Brockmüller, O. Günlük, and L. Wolsey. "Designing private line networks - polyhedral analysis and computation". Discussion Paper 9647, Center for Operations Research and Econometrics, October 1996.

[21] B. Brockmüller, O. Günlük, and L. Wolsey. "Designing private line networks - polyhedral analysis and computation". Discussion Paper 9647 revised, Center for Operations Research and Econometrics, March 1998.

[22] G. Cho and D.X. Shaw. "A depth-first dynamic programming algorithm for the tree knapsack problem". Research Memorandum 94-15, School of Industrial Engineering, Purdue University, April 1994.

[23] G. Cho and D.X. Shaw. "Limited column generation for local access telecommunication network design - formulations, algorithms, and implementation". Working Paper, January 1995.

[24] T. Christof and G. Reinelt. "Algorithmic aspects of using small instance relaxations in parallel branch-and-cut". Working paper, April 1998.

[25] L.W. Clarke and P. Gong. "Capacitated network design with column generation". Working paper, December 1995.

[26] Y. Crama. "Combinatorial optimization models for production scheduling in automated manufacturing systems". In *14th European Conference on Operations Research*, pages 237–259, 1995.

[27] H. Crowder, E. Johnson, and M. Padberg. "Solving large-scale zero-one linear programming problems". *Operations Research*, 31(5):803–834, 1983.

[28] G. Dahl, A. Martin, and M. Stoer. "Routing through virtual paths in layered telecommunication networks. Research note n 78/95, Telenor Research and Development, Kjeller, Norway, 1995.

[29] G. Dahl and M. Stoer. "A cutting plane algorithm for multicommodity survivable network design problems". Working Paper, April 1996.

[30] E. Dijkstra. "A note on two problems in connexion with graphs". *Numeriche Mathematiks*, 1:269–271, 1959.

[31] R. Epstein. Linear programming and capacitated network loading, 1998.

[32] S. Even, A. Itai, and A. Shamir. On the complexity iof timetable and multicommodity flow problems. *SIAM Journal of Computing*, 5:691–703, 1976.

[33] O. Flippo, A. Kolen, A. Koster, and R. van de Leensel. A dynamic programming algorithm for the local access network expansion problem. Research Memorandum 96/027, Maastricht University, 1996.

[34] K. Fukuda. "cdd+ reference manual". Technical report, Institute for Operations Research, ETH-Zentrum, Zurich, Switzerland, 1995.

[35] V. Gabrel and M. Minoux. "Large scale lp relaxations for minimum cost multicommodity flow problems with step increasing cost functions and computational results". Technical Report Masi 96/17, Laboratoire d'Informatique de Paris 6, June 1996.

[36] V. Gabrel and M. Minoux. "Lp relaxations better than convexification for multicommodity network optimization problems with step increasing cost functions". *ACTA Mathematica Vietnamica*, 22:123–145, 1997.

[37] M. R. Garey and D.S. Johnson. *"Computers and intractability: a guide to the Theory of NP-Completeness"*. Freeman and Company, N.Y., 1979.

[38] B. Gavish. "Topological design of telecommunication networks: Local access design networks". *Annals of Operations Research*, 33:17–71, 1991.

[39] B. Gavish. Topological design of computer communication networks - the overall design problem. *European Journal of Operational Research*, 58(2):149–172, 1992.

[40] B. Gavish and K. Altinkemer. "Backbone network design tools with economic tradeoffs". *ORSA Journal on Computing*, 2(3):236–252, 1990.

[41] D.E. Goldberg. *"Genetic Algorithms in Search, Optimization and Machine Learning"*. Addison Wesley, Ma., 1989.

[42] M. Grötschel and C. Monma. "Integer polyhedra arising from certain network design problems with connectivity constraints". *SIAM Journal on Discrete Mathematics*, 3(4):502–523, 1990.

[43] M. Grötschel, C. Monma, and M. Stoer. "Computational results with a cutting plane algorithm for designing communication networks with low-connectivity constraints". *Operations Research*, 40(7):309–330, 1992.

[44] M. Grötschel, C. Monma, and M. Stoer. "Polyhedral and computational investigations for designing communication networks with high survivability requirements". *Operations Research*, 43(6):1012–1024, 1995.

[45] P.L. Hammer, E.L. Johnson, and U.N. Peled. "Facets of regular 0-1 polytopes". *Mathematical Programming*, 8:179–206, 1975.

[46] D. Hartvigsen and E. Zemel. "The complexity of lifted inequalities for the knapsack problem". *Discrete Applied Mathematics*, 39:113–123, 1992.

[47] S.H. Hwan and W. Shogan. "Modeling and solving an FMS part selection problem". *International Journal of Production Research*, 27:1349–1366, 1989.

[48] CPLEX Optimization Inc. Cplex callable library, version 4.0, 1995.

[49] ITU-T. Com xviii 228-e, March 1984. Geneva.

[50] C. Jack, S-R. Kai, and A. Shulman. "Design and implementation of an interactive optimization system for telephone network planning". *Operations Research*, 40:14–25, 1992.

[51] D.S. Johnson and K.A. Niemi. "On knapsacks, partitions and a new dynamic programming technique for trees". *Mathematics of Operations Research*, 8:1–14, 1983.

[52] A. Kolen. A genetic algorithm for frequency assignment. Technical report, Maastricht University, 1999. Available at http://www.unimaas.nl/~akolen/.

[53] A. Lisser, R.Sarkissian, and J.P.Vial. "Survivability in telecommunication networks". Technical report, France Telecom, CNET, 1995.

[54] F. Maffioli M. Dell'Amico and S. Martello. *Annotated Bibliographies in Combinatorial Optimization*. Wiley, New York, 1997.

[55] T.L. Magnanti, P. Mirchandani, and R. Vachani. "The convex hull of two core capacitated network design problems". *Mathematical Programming*, 60:223–250, 1993.

[56] T.L. Magnanti, P. Mirchandani, and R. Vachani. "Modelling and solving the two-facility capacitated network loading problem". *Operations Research*, 43:142–157, 1995.

[57] J.W. Mamer and W. Shogan. "A constrained capital budgeting problem with applications to repair kit selection". *Management Science*, 33:800–806, 1987.

[58] H. Marchand and L.A. Wolsey. "The 0-1 knapsack problem with a single continous variable". Discussion Paper 9720, Center for Operations Research and Econometrics, March 1997.

[59] H. Marchand and L.A. Wolsey. "Aggregation and Mixed Integer Rounding to solve MIPs". Discussion Paper 9839, Center for Operations Research and Econometrics, June 1998.

[60] C. Monma, B. Munson, and W. Pulleyblank. "Minimum-weight two-connected spanning networks". *Mathematical Programming*, 46:153–171, 1990.

[61] C. Monma and D. Shallcross. "Methods for designing communications networks with certain two-connectivity constraints". *Operations Research*, 37:531–541, 1989.

[62] G.L. Nemhauser and L.A. Wolsey. *"Integer and Combinatorial Optimization"*. Wiley, N.Y., 1988.

[63] I. Osman and G. Laporte. "Metaheuristics: a bibliography". *Annals of Operations Research*, 63:513–623, 1996.

[64] K. Park and S. Park. Lifting cover inequalities for the precedence-constrained knapsack problem. *Discrete Applied Mathematics*, 72:219–241, 1972.

[65] U. Paul, P. Jonas, D. Alevras, M. Grötschel, and R. Wessäly. "Survivable mobile phone network architectures: Models and solution methods". Preprint SC 96-48, Konrad-Zuse-Zentrum fur Informationstechnik Berlin, 1996.

[66] H. Pirkul and S. Narasimhan. "Primary and secondary route selection in backbone computer networks". *ORSA Journal on Computing*, 6:50–60, 1994.

[67] Y. Pochet and L.A. Wolsey. Integer knapsack and flow covers with divisible coefficients: Polyhedra, optimization and separation. *Discrete Applied Mathematics*, 59(1):57–74, 1995.

[68] A. Schrijver. *Theory of linear and integer programming.* Wiley, New York, 1986.

[69] D.X. Shaw. "Reformulation and column generation for several telecommunication network design problems". In *Proceeding of the 2nd International Telecommunication Conference*, Nashville, Tennessee, 1994.

[70] D.X. Shaw and G. Cho. "A branch-and-bound procedure for the tree knapsack problem". Research memorandum 94-11, School of Industrial Engineering, Purdue University, March 1994.

[71] A. Shulman and R. Vachani. "An algorithm for capacity expansion of local access networks". In *IEEE Infocom'90*, San Francisco, California, 1990.

[72] K.E. Stecke and I. Kim. "A study of part type selection approaches for short-term production planning". *International Journal of Flexible Manufacturing Systems*, 1:7–29, 1988.

[73] R.E. Tarjan. *"Data structures and Network Algorithms"*. SIAM, Philadelphia, Pa, 1983.

[74] S. Thienel. Abacus: a branch and cut system, version 1.2, 1996.

[75] R.L.M.J. van de Leensel, O.E. Flippo, and A.M.C.A. Koster. "A dynamic programming algorithm for the ATM network installation problem on a tree". Research Memorandum 98/009, Maastricht University, 1998.

[76] R.L.M.J. van de Leensel, C.P.M. van Hoesel, and J.J. van de Klundert. "Lifting Valid Inequalities for the Precedence Constrained Knapsack Problem". Research Memorandum 97/021, Maastricht University, 1997. To appear in Mathematical Programming.

[77] C.P.M. van Hoesel, A.M.C.A. Koster, R.L.M.J. van de Leensel, and M.W.P. Savelsbergh. Polyhedral results for the edge capacity polytope. Technical report, Maastricht University, 1999.

[78] L. Wolsey. Facets and strong inequalities for integer programs. *Operations Research*, 24:367–372, 1976.

[79] L.A. Wolsey. "Faces of linear inequalities in 0-1 variables". *Mathematical Programming*, 8:165–178, 1975.

[80] E. Zemel. "Easily computable facets of the knapsack polytope". *Mathematics of Operations Research*, 14(4):760–764, 1989.

Nederlandse Samenvatting

Telecommunicatie netwerken bevinden zich in een dynamische omgeving. Sinds de introductie van de telefonie hebben de markten een continue groei in de vraag naar telecommunicatie diensten geregistreerd. Daarnaast hebben technologische ontwikkelingen een belangrijke bijdrage geleverd aan de omvang en het ontwerp van netwerken. Enerzijds heeft dit geleid tot wereldwijde communicatie faciliteiten en de introductie van nieuwe telecommunicatie diensten. Anderzijds hebben deze innovaties in sterke mate bijgedragen aan de profatibiliteit van de telecommunicatie industrie. Tenslotte heeft de deregelering heeft gezorgd voor concurrerende markten, welke ondernemingen stimuleren om efficient te opereren.

Telecommunicatie netwerken worden voortdurend aangepast aan deze veranderende omstandigheden. Gedurende dit proces worden talrijke beslissingen genomen welke van invloed zijn op de karakteristieken van een netwerk, zoals de capaciteit, betrouwbaarheid, beschikbaarheid, en niet in de laatste plaats de kosten. Elk jaar worden er wereldwijd immense bedragen geinvesteerd in het onderhoud en moderniseren van telecommunicatie netwerken. Dientengevolge ontstaan er ook talloze mogelijkheden tot kostenbesparingen. De problemen die opgelost dienen te worden om zulke kostenreducties te bewerkstelligen zijn vaak complex. Wiskundige technieken uit het wetenschappelijke werkveld van de besliskunde kunnen een helpende hand bieden bij de analyse van dergelijke netwerk ontwerp problemen.

Dit proefschrift beschrijft zowel modellen als methoden voor een aantal specifieke netwerk ontwerp problemen. **Hoofdstuk 1** geeft een inleiding op het algemene ontwerp van telecommunicatie netwerken en introduceert de desbetreffende terminologie. De meest gangbare netwerken hebben een hierarchisch ontwerp. De onderste lagen van zulk een netwerkstructuur zijn verantwoordelijk voor de aansluiting van telecommunicatie klanten aan het netwerk. De hogere lagen verzorgen de verbinding tussen lokale aansluitnetwerken en de routering van telecommunicatie boodschappen tussen de gebruikers van een dienst. Het inleidende hoofdstuk beschrijft zowel een situatie welke de problematiek in de lokale aansluitnetwerken representeert, als een voorbeeld welke de beslissingsmogelijkheden in routeringsnetwerken illustreert. Tevens dient het hoofdstuk als een leidraad voor de rest van het proefschrift, welke is opgesplitst in twee delen.

Part I bestaat uit de hoodstukken 2, 3 en 4, en behandelt lokale aansluitnetwerken. **Hoofdstuk 2** introduceert een netwerk probleem waarbij onder invloed van toenemende vraag naar telecommunicatie diensten de bestaande capaciteit in een lokaal aansluit-

netwerk niet meer toereikend is om in de gewenste service te kunnen voorzien. Voor de noodzakelijke uitbreiding van capaciteit onderscheiden we twee mogelijkheden. Enerzijds zorgt de aanleg van additionele kabels in de bestaande netwerkstructuur voor een grotere capaciteit. Anderzijds kunnen centrales in het lokale aansluitnetwerk geplaatst worden welke telecommunicatie stromen kunnen comprimeren. Aangezien gecomprimeerde telecommunicatie stromen gepaard gaan met een kleinere consumptie van capaciteit leiden dergelijke centrales tot een vermindering van de benodigde capaciteit. Kern van het probleem is om een efficiente afweging te maken tussen de kosten van kabelexpansie en kosten van installatie van centrales. Hoofdstuk 2 beschrijft een elegante methode voor het vinden van de optimale capaciteitsexpansie van een lokaal aansluitnetwerk. De methode blijkt in staat om probleeminstanties uit de telecommunicatie industrie efficient op te lossen. Als zodanig kan de methode een belangrijke rol spelen in het economisch plannen van lokale aansluitnetwerken.

In **hoofdstuk 3** beschouwen we een uitbreiding van de problematiek welke onderdeel is van hoofdstuk 2. Een gedeelte van de telecommunicatie stromen in een lokaal aansluitnetwerk is afkomstig van communicatie tussen klanten in hetzelfde lokale aansluitnet. Communicatie tussen elk tweetal gebruikers van een dienst kan alleen plaatsvinden via tussenkomst van een routeringscentrale. Deze routeringscentrales bevinden zich normaal gesproken in de hogere lagen van de netwerk hierarchie. Dit betekent dat, ondanks dat gebruikers in eenzelfde lokaal aansluitnetwerk geografisch dicht bij elkaar zitten, de communicatie tussen desbetreffende gebruikers een lange weg in het telecommunicatie netwerk af kan leggen. In hoofdstuk 3 beschouwen we daarom de additionele mogelijkheid om dergelijke routeringscentrales in het lokale aansluitnetwerk te installeren om zodanig deze routering eenvoudiger te maken. De installatie van een routeringscentrale leidt zodoende tot een vermindering van de capaciteitsconsumptie en derhalve is zij concurrerend met andere maatregelen om capaciteitsproblemen in lokale aansluitnetwerken op te lossen. We beschijven in dit hoofdstuk een methode welke de juiste balans vindt tussen de kosten van kabelexpansie en de kosten van installatie van centrales. Probleeminstanties uit de praktijk, zoals beschikbaar gesteld door KPN Research, Leidschendam, kunnen hiermee efficient worden opgelost.

Veel netwerk ontwerp problemen op lokale aansluitnetwerken hebben een gemeenschappelijke deler. Ten eerste is de netwerkstructuur zodanig dat er tussen elk tweetal klanten in eenzelfde aansluitnetwerk precies één uniek pad is. Dit impliceert dat routering van telecommunicatieberichten binnen het lokale aansluitnet in het algemeen geen keuzemogelijkheden met zich mee brengt. Ten tweede geldt dat de centrales welke in een lokaal aansluitnet aanwezig zijn of geinstalleerd kunnen worden vaak beschikking hebben over een beperkte capaciteit, zodat een keuze moet worden gemaakt welke klanten aangesloten worden op een specifieke centrale. Tenslotte worden meestal beperkende voorwaaarden opgelegd aan de toewijzing van klanten aan centrales met het doel een bepaalde logische en overzichtelijke netwerkstructuur te ontwerpen. Inzichtelijkheid is een belangrijke factor voor een efficiente planning van bijvoorbeeld onderhoud. In **hoofdstuk 4** beschouwen we de mathematische formulering welke de gemeenschappelijke deler van deze problemen in lokale aansluitnetwerken representeert. We bestuderen versterkingen van de formulering, de complexiteit van zulke verbeteringen en het daaruitvolgende effect.

Part II, bestaande uit hoofdstukken 5 en 6, beschouwt telecommunicatie netwerk ontwerp problemen in hogere lagen van de hierarchische structuur. We beschouwen de lokatie van routeringscentrales hierbij als een gegeven, en concentreren ons op de beslissingen omtrent de routering van telecommunicatie stromen en de installatie van capaciteit op de connecties tussen centrales. **Hoofdstuk 5** introduceert een aantal wiskundige modellen en heuristische methoden voor deze *network loading* problemen. Tevens beschrijven we enkele kenmerken van een beslissingsondersteunend computer systeem welke in samenwerking met KPN Research te Leidschendam ontwikkeld is voor de efficiente analyse van deze netwerk problemen. De besproken software bevat een grafische interface welke het gebruik en de interpretatie van netwerk planningen beduidend eenvoudiger maakt.

Een belangrijk onderdeel van de modellen voor de network loading problemen zoals beschreven in hoofdstuk 5 wordt gevormd door capaciteitsrestricties. Deze restricties garanderen dat de hoeveelheid capaciteit welke geinstalleerd wordt op een connectie in het netwerk groter of gelijk is aan de capaciteitsbehoefte op de desbetreffende connectie. In **hoofdstuk 6** restricteren we ons tot één specifieke connectie in het netwerk en analyseren de mathematische structuur van de bijbehorende capaciteitsrestrictie. Verschillende theoretische resultaten worden vermeld. Ten eerste wijzen we op het nut van versterkingen van individuele capaciteitsrestricties voor globale network loading problemen. Vervolgens laten we zien hoe reeds bekende ongelijkheden voor knapsack problemen kunnen worden gebruikt om versterkingen voor het onderhavige model te genereren. Meer specifiek concentreren we ons op twee typen versterkingen welke een grafische interpretatie hebben. Het effect van deze versterkingen op de solvabiliteit van network loading problemen is het onderwerp van een rekenstudie.

Curriculum Vitae

Robert van de Leensel was born on March 21st, 1971 in Helenaveen, the Netherlands. He studied Econometrics at the Katholieke Universiteit Brabant in Tilburg, specializing in Operations Research. During his study he tutored mathematics and statistics classes for two years, spent six months at North Carolina State University in Raleigh, USA, and participated in a Master's course in Management Science at CentER, Tilburg. After his graduation in April 1995 he started his PhD research at Maastricht University, where he completed his Doctoral thesis.

Robert van de Leensel werd geboren op 21 maart 1971, te Helenaveen. Hij studeerde Econometrie aan de Katholieke Universiteit Brabant te Tilburg, met als specialisatie Besliskunde. Gedurende zijn studie was hij twee jaar werkzaam als student-assistent in de vakken wiskunde en statistiek, verbleef hij een halfjaar aan North Carolina State University in Raleigh, USA, en volgde hij een Master's course in Management Science aan het CentER, te Tilburg. Na zijn afstuderen (cum laude) in april 1995 begaf hij zich naar de Universiteit Maastricht, alwaar hij zijn proefschrift voltooide.

Robert van de Leensel was born on March 21st, 1971 in Helenaveen, the Netherlands. He studied Econometrics at the Katholieke Universiteit Brabant in Tilburg, specializing in Operations Research. During his study he tutored mathematics and statistics classes for two years, spent six months at North Carolina State University in Raleigh, USA, and participated in a Master's course in Management Science at CentER, Tilburg. After his graduation in April 1995 he started his PhD research at Maastricht University, where he completed his Doctoral thesis.

Robert van de Leensel werd geboren op 21 maart 1971, te Helenaveen. Hij studeerde Econometrie aan de Katholieke Universiteit Brabant te Tilburg, met als specialisatie Besliskunde. Gedurende zijn studie was hij twee jaar werkzaam als student-assistent in de vakken wiskunde en statistiek, verbleef hij een halfjaar aan North Carolina State University in Raleigh, USA, en volgde hij een Master's course in Management Science aan het CentER, te Tilburg. Na zijn afstuderen (cum laude) in april 1995 begaf hij zich naar de Universiteit Maastricht, alwaar hij zijn proefschrift voltooide.

ISBN 90-9012781-X