

Capacity Expansion in Contemporary Telecommunication Networks

by

Raghavendran Sivaraman

Submitted to the Sloan School of Management
in partial fulfillment of the requirements for the degree of

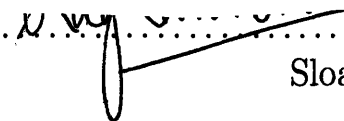
Doctor of Philosophy in Operations Research

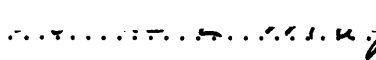
at the


MASSACHUSETTS INSTITUTE OF TECHNOLOGY

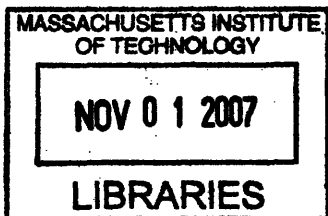
September 2007

© Massachusetts Institute of Technology 2007. All rights reserved.

Author 
Sloan School of Management
August 08, 2007

Certified by 
Thomas L. Magnanti
Institute Professor
Thesis Supervisor

Accepted by 
Dimitris Bertsimas
Boeing Professor of Operations Research
Co-director, Operations Research Center



ARCHIVES

Capacity Expansion in Contemporary Telecommunication Networks

by

Raghavendran Sivaraman

Submitted to the Sloan School of Management
on August 08, 2007, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Operations Research

Abstract

We study three capacity expansion problems in contemporary long distance telecommunication networks.

The first two problems, motivated by a major long distance provider, address capacity expansion in national hybrid long distance telecommunication networks that use both the traditional TDM technology and more recent VoIP technology to transport voice calls. While network capacity expansion in general is known to be hard to approximate, we exploit the unique requirements associated with hybrid networks to develop compact models and algorithms with strong performance guarantees for these problems.

For a single period single facility capacity expansion problem in a hybrid network, using a decomposition approach we design a $(2 + \epsilon)$ -factor approximation algorithm. Generalizing this idea, we propose a Decentralized Routing Scheme that can be used to design approximation algorithms for many variations of hybrid network capacity expansion.

For the Survivable Capacity Expansion Problem in hybrid networks, in which we are required to install enough capacity to be able to support all demands even if a single link fails, we propose a compact integer program model. We show that this problem is APX-Hard, and present two heuristics with constant worst case performance guarantees.

Finally, we consider the capacity planning problem when peak demands occurring at different times can share network capacity. We propose a general model for capturing time variation of demand, and establish a necessary and sufficient condition for a capacity plan to be feasible. Using a cutting plane approach, we develop a heuristic procedure. Computational experiments on real and random instances show that the proposed procedure performs well, producing solutions within 12% of optimality on average for the instances tested. The tests also highlight the significant savings potential that might be obtained by capacity planning with time sharing.

Thesis Supervisor: Thomas L. Magnanti

Title: Institute Professor

Acknowledgments

My stay at MIT has been the most enriching experience of my life till now. I express my gratitude to all the people who have made this place great.

Foremost, I thank my advisor Tom Magnanti for taking the time to advise me when he was Dean. He was always encouraging, open-minded, flexible, and put my interests ahead of everything else. I benefited immensely from his clarity of thought and his emphasis to keep things as simple as possible. I learnt a lot from Tom, not just academically, but also about life. I feel very lucky to have been his student and I hope that a fraction of his humility and infinite patience will rub off on me.

I thank the other members of my committee, Jim Orlin and Steve Graves, for many useful suggestions. My discussion with them motivated the Capacity Sharing problem addressed in this thesis. Special thanks to Jim for the proof of APX-Hardness for the Hub-and-Spoke Capacity Expansion Problem.

I am grateful to Verizon Laboratories for funding part of this research. In the three summers I spent at the Labs I was able to apply what I learnt to real problems, and gained practical experience of great value to me. I thank Rina Schneur for being a very understanding boss and providing a supportive work environment. Don Smith, Roger Tobin, Hui Liu and Larry Crom were all great people to work with, and made my stints at Verizon pleasurable.

I thank Martin Skutella for inviting me to work with him at Max Planck Institut at Saarbruecken and teaching me the basics of doing research.

I thank G Srinivasan for introducing me to Operations Research. Passionate, approachable, and a great teacher, he was the main reason for my initial interest in OR. I am glad I took his courses.

I thank Lisa Bleheen, not only for giving me access to Tom, but also for making me feel comfortable when I wait at the Dean's Office. She constantly enquired about me and my family, and was possibly more concerned than me about how my wife was coping with her new life in the US. Thanks also to Laura Rose, Paulette Mosley, and Andrew Carvalho who run the ORC more efficiently than any other place I have

seen. They also look out for all the students, and make sure that the students are comfortable with their academics as well as their social life at the ORC.

I thank Dan, Shobhit, Pranava, Pavithra and many other friends at the ORC who made the place much more than just an office. Thanks also to Suri, for sharing his wisdom on all topics under the sun, and much more importantly for inviting me for great home cooked food many many times. I thank all my other friends at MIT, in particular, Vikram, Hari, Srini, Sivaram, jSrini, Bhanu and Vikrant, with whom I played cricket, badminton and monopoly, acted in skits for Diwali, and went skiing. The last five years would not have been half as delightful without them.

Finally, I thank my family for their unconditional love and support. I thank Appa for constantly encouraging me to try new things and instilling in me the ambition to do better. I thank Amma for always believing that I could do a lot more than I actually can. Thanks also to my brothers Vaidy and Bala for their sense of humor and never failing to make me laugh. My deepest gratitude goes to my wife, Anu, for always being there for me, in spite of not being with me during most of my stay here. She showed infinite patience, especially in the last eight months when she was here with me. Getting used to a new country and making new friends is tough by itself, without also having to put up with me working late and during weekends. She was very understanding and never complained (for the most part!), and for this I am very grateful.

Contents

1	Introduction	15
1.1	Motivation	16
1.1.1	Hybrid Network Capacity Expansion	16
1.1.2	Network Planning With Capacity Sharing	18
1.2	Organization Of The Thesis	20
1.3	Preliminaries	22
1.4	Literature review	25
1.4.1	Capacity Expansion Problem	25
1.4.2	Survivable Capacity Expansion Problem	27
1.4.3	Network Planning with Capacity Sharing	28
2	Single Period Capacity Expansion	31
2.1	Modeling the Capacity Expansion Problem	32
2.2	Complexity Results	36
2.3	The Uncapacitated Problem	39
2.3.1	An Efficient Primal-Dual Algorithm	41
2.4	Hub-and-Spoke Networks Without Initial Capacities	42
2.4.1	A Formulation For HSP Without Initial Capacities	43
2.4.2	A Constant Factor Approximation Algorithm	45
2.4.3	Asymptotic Optimality of the Rounding Method	50
2.5	An Approximation Algorithm For The General Case	51
2.5.1	HSP With Small Initial Capacities	51
2.5.2	Arbitrary Initial Capacities	55

2.5.3	A Tight Example	60
3	Extensions of Hybrid Network Capacity Expansion Problem	63
3.1	The Decentralized Routing Scheme	64
3.2	Upper Bounds On The Number Of Facilities	68
3.2.1	Routing Demands To Softswitches	68
3.2.2	Bounds On The Number Of Facilities On Direct Links	69
3.2.3	A Model For CEP With Upper Bounds	70
3.2.4	A Decentralized Routing Algorithm For CEP With Upper Bounds	71
3.3	Unsplittable Demands	73
3.3.1	The Expandable Min-Knapsack Problem	75
3.3.2	Multiple Facility Types	76
4	Capacity Expansion with Single Link Failure Protection	79
4.1	Modeling the SCEP in a Hybrid Network	80
4.2	Parallel Path Network Restoration Problem	85
4.3	Lower Bounds For The SCEP	91
4.4	Decentralized Routing Algorithms for Survivable Capacity Expansion	93
4.4.1	A $(5 + \epsilon)$ -approximation Algorithm	94
4.4.2	A $(4 + \epsilon)$ -approximation Algorithm	104
5	Network Planning With Capacity Sharing	113
5.1	The Capacity Sharing Problem	113
5.2	Modeling Time Variations of Demand	116
5.3	Capacity Sharing: Modeling and Complexity	120
5.3.1	A Condition For Feasibility	120
5.3.2	The Separation Problem	122
5.3.3	When Cut Constraints Are Sufficient	124
5.4	Solving The Capacity Sharing Problem	128
5.4.1	Single Demand Optimization	129
5.4.2	Fractional Flow Variables	130

5.4.3	A Cutting Plane Heuristic By Partitioning Commodities . . .	132
6	Computational Experiments	139
6.1	Algorithm Summary and Implementation	139
6.2	Test Problems	140
6.3	Computational Results	142
7	Conclusions	147
A	APX Hardness Proof for CEP in Hybrid Networks	151

List of Figures

1-1	Dynamic Routing. (<i>Source:</i> Clark [19])	20
1-2	A telecommunication network	23
2-1	A hub-and-spoke network with direct links	35
2-2	Reduction of PARTITION to SNLP	37
2-3	A bad example for the Decentralized Routing Algorithm	61
4-1	A star network with parallel radial links	80
4-2	The L-Reduction for the SCEP	84
4-3	A two-node parallel edge network	85
4-4	Slope used by the binary search algorithm	88
4-5	Optimal cost of the network restoration problems with upper and lower bounds	100
5-1	Reduction from Set Cover Problem	123
5-2	Demand graphs for which cut conditions are sufficient. (a) K_4 (b) C_5 (c) Union of two stars.	125
5-3	The original demand variation and the approximation by the partition $\{d_1, d_2\}, \{d_3\}$	127
5-4	The Single Stage Model	129

List of Tables

6.1	Real problem instances	141
6.2	Computational results for real problem instances	143
6.3	Computational results for real networks with random demand	144
6.4	Computational results for random 10 node networks	144
6.5	Computational results for random 20 node networks	145
6.6	Computational results for random 30 node networks	145

Chapter 1

Introduction

We study capacity expansion problems that are of interest to a contemporary telecommunications service provider. It is hard to overemphasize the importance of optimal long-distance capacity planning in the highly competitive telecommunication industry, especially at a time when prices of long-distance services have been steadily declining. And understandably, the literature abounds with models and methods for a wide variety of network design and capacity planning problems. We add to this body of work by studying three important capacity expansion problems that arise in today's long distance telecommunication networks.

The first two problems we address are capacity expansion problems in a special but now common kind of long-distance telecommunication networks called hybrid networks. Hybrid long-distance networks use both the traditional TDM technology and the more recent VoIP technology to transport voice calls. Capacity expansion in a hybrid long-distance network has unique requirements, and we develop models and algorithms that address these requirements, and at the same time, exploit the problem's special structure.

The third problem we study seeks to identify a minimum cost capacity expansion plan that explicitly accounts for capacity sharing possibilities across origin-destination pairs. In a long-distance network spanning the entire country, peak demands between locations do not all occur at the same time. Today's switching systems allow flexible routing of calls with the possibility of using excess capacity in a different part of a

network to route a call. We model this variation in peak demands across the network and develop an algorithm to produce a capacity plan that (heuristically) minimizes expansion cost.

The main focus of this thesis is to develop models and fast heuristic solution methods for these problems. Since capacity expansion in general is hard to solve, and exact algorithms for these problems usually have large running times, there is a need for tractable near optimal heuristics. We establish a priori guarantees on the quality of the solution for some of our algorithms. We also develop compact integer programming formulations that exploit the special structure of these problems that can be solved as such when the network size is small.

While the primary motivation for the work in this thesis is telecommunication network planning, we believe that some of the models and algorithms might be applicable in other contexts. We model some of our problems as optimization on hub and spoke networks. The ubiquity of hub and spoke networks in airlines and logistics suggests that our models might be applicable in those domains. Particularly, we believe that our models might be applied to airline network design, and to the logistics of package shipments.

1.1 Motivation

1.1.1 Hybrid Network Capacity Expansion

The introduction of Voice over Internet Protocol (VoIP) in the 1990s marked a major technological shift in the telecommunication industry. Voice calls are traditionally routed using circuit switching by reserving a dedicated path for every call in progress. Many voice streams are combined using Time Division Multiplexing (TDM) to be transported between switches in a long distance network. VoIP, on the other hand, uses packet switching: voice is converted into data packets and routed across the network using the Internet Protocol. VoIP has emerged as a significantly cheaper alternative to TDM based switching in long-distance telecommunication networks

while matching it in quality, reliability and security. VoIP is preferred by telecommunications providers for a number of reasons (see, for example, Dodd [21]): VoIP equipment has a lower cost; packet-switched networks more easily offer many value added services like voicemail, messaging, and internet access; and the provider can exploit economies of scale by using the same underlying network to transport voice, data, and video.

The idea of using the same network for both voice and data is referred to as the convergence of voice and data. According to Green [25], “If the telecommunication experts are unanimous on anything, it is the conviction that all forms of communication are gravitating toward a single unified IP network.” For a detailed analysis of the advantages of convergence and the barriers, see Green [25].

While telecommunication service providers would like to use a converged IP network, the transition to this technology cannot be abrupt since the companies have substantial investments in the existing circuit switched network. The softswitch is a technology that makes a smooth and gradual migration possible. It communicates with a TDM switch using circuit switching, while it uses packet switching to communicate with other softswitches. In addition to aiding convergence, softswitches also have other advantages. In particular, they reduce capital costs as well as operating expenses. As a result, most current long-distance networks are hybrids containing both legacy (TDM) switches and softswitches.

We study the Capacity Expansion Problem (CEP) in these hybrid networks, seeking to determine the amount of additional capacity to install on each trunk of the network to be able to route a given demand forecast. While the CEP is well studied for general networks, hybrid networks have special characteristics that we exploit to develop compact models as well as heuristic procedures with strong performance bounds.

Reliability of the network and high quality of service have a direct impact on customer retention, and consequently on the profitability of a telecommunications service provider. In the telecommunication industry, a typical quality of service goal is to have less than 1 percent blocked calls. However, failures in the network suddenly

reduce available capacity, and therefore could result in significant degradation in the quality of service. A common approach for ensuring reliable service is to build a network that can maintain an acceptable quality of service even when some part of the network fails. Wang [53] and Ash [4] provide a more detailed introduction to the importance of failure protection in a telecommunication network including instances of network failure and some technologies to handle them.

Network planners need to decide which kind of failures the network needs to be protected against, which we call survivability requirements. Since the cost of capacity expansion depends on the survivability requirements, planners must make a tradeoff between the level of protection and the cost of expansion. They make this decision based on the likelihood of the different failures and the associated cost of protection. A commonly used survivability criterion is protection against single link failures. When a link fails, all the capacity on that link becomes unavailable. The motivation for using single link failure protection is the assumption that failures are rare, that only one link fails at a time, and the failed link can be repaired before another fails. We study the problem of expanding capacity in a hybrid network when protecting against single link failures.

1.1.2 Network Planning With Capacity Sharing

Traditionally, telecommunication network planners use a single demand forecast for planning the network. A popular demand estimate is known as the busy hour peak demand, calculated as the average number of calls that originate in the busiest hours of the ten busiest days of the year (see Clark [19] for more details). This forecast is also adjusted based on the outlook for demand growth.

However, the demand forecast for each origin-destination pair is arrived at independent of the other O-D pairs. Consequently, when sizing the network, service providers plan for the peak demands of all the origin destination pairs occurring simultaneously. However, the busiest days of the year, and certainly the busiest hours, for origin-destination pairs in different geographical locations do not necessarily coincide.

Clark [19] states that “In practice, telecommunication networks are found to have a

discernible busy hour” (for every origin destination pair). This busy hour depends on the type of area (business or residential), time-zone of the locations, and any time-zone differences between the origin and destination. For example, for a residential location, busy hours usually occur in the evening, while business areas typically incur two busy hours, one in the morning and another in the afternoon. Clark also observes that traffic between locations in significantly different time zones (mainly international calls) have a distinct peak time since they are limited by the time-zone difference. In addition to intra-day traffic patterns, locations might have different busiest days of the year because of large scale local festivals and other events. Therefore, for the network as a whole, it is reasonable to expect a significant variation in the busy hours of different origin-destination pairs.

Capacity Sharing is made possible by routers that employ Dynamic Routing (see the books by Clark [19] and Ash [5]). In Dynamic Routing, call routes are not fixed but change with the time of day as well as the current status of the network: the number of calls between origin-destination pairs and the load on different trunks. This capability permits providers to use capacity in a part of the network that currently has less load to service calls of O-D pairs that have fully utilized their primary routes and leads to better utilization of capacity across the network.

Clark [19] presents the following illustration of dynamic routing which we describe. In Figure 1-1, calls between New York and Washington DC are routed primarily along the direct trunk between the two cities. However, early in the morning (in the eastern time zone), when the demands out of the west are low, (if required) dynamic routing automatically supplements the New York to Washington DC capacity of the network by routing some of these calls through Los Angeles. Later in the day, when traffic from Los Angeles increases, the system no longer uses the additional route to supplement capacity between New York and Washington DC.

AT&T implemented Dynamic Routing as early as 1987. Ash and Oberer [6] describe the implementation and measure the benefits of Dynamic Routing by evaluation network blocking probabilities (fraction of calls not routed) on some of the busiest days of the year including Christmas, Thanksgiving, Mothers day and Easter.

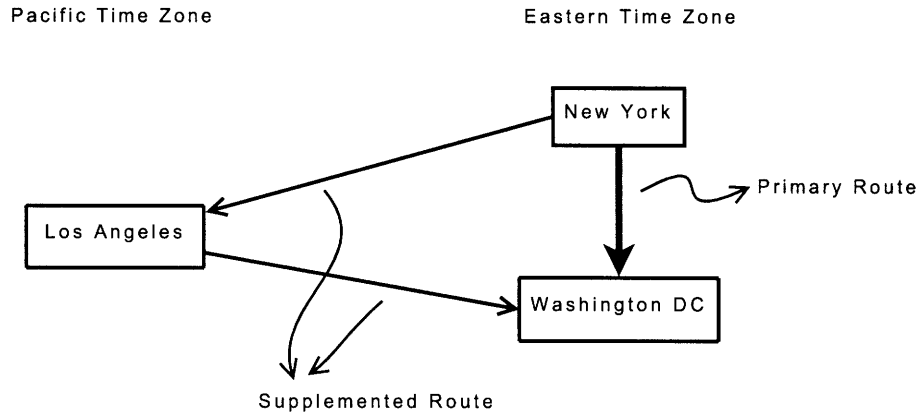


Figure 1-1: Dynamic Routing. (*Source: Clark [19]*)

Since Dynamic Routing will employ unused capacity in the network to route calls, from a network planning point of view, we need to ensure that we install enough capacity on the trunks of the network so that for any given traffic pattern across the network, and not just on any single trunk, there is a feasible routing of calls. This is the Capacity Sharing problem that we study in this thesis.

One of the important aspects of this problem is how the set of all traffic patterns (which we call the demand variation) that the network will be designed to meet. We describe few different models for demand variations, and consider a general model that assumes the set of all demand vectors belong to a polyhedron. In addition to establishing theoretical results for the problem (for example, necessary and sufficient conditions for a capacity plan to be feasible), we present a heuristic procedure that could be used to find near optimal capacity augmentation plans for the network, thereby leading to significant savings when compared to designing the network for all individual peak demands.

1.2 Organization Of The Thesis

The remainder of this chapter is organized as follows. In Section 1.3, we introduce telecommunication network related terminology as well as a few well known concepts and results that we use in this thesis. Section 1.4 provides a review of literature

relevant to each of the three problems that we study.

Chapter 2 addresses the Single Period Single Facility Capacity Expansion Problem (CEP) in a hybrid network. We describe unique requirements associated with a hybrid network, and present a model for the Capacity Expansion Problem that exploits these requirements. We show that this problem is NP-Hard for very restrictive special cases. However, when the problem is uncapacitated (that is, when the capacity is installed in very large units), the problem is polynomially solvable. We then present a 2-approximation algorithm for the CEP when the network has no initial capacity. The main result of the chapter is a $(2 + \epsilon)$ -approximation algorithm for any positive ϵ for the CEP in its most general form.

In Chapter 3, we extend the idea used to design an approximation algorithm for the CEP to variants of the CEP that address important practical constraints. We refer to this general framework, which is applicable to a variety of hybrid network Capacity Expansion Problems, as a Decentralized Routing Scheme. We then use this scheme to obtain constant factor approximation algorithms for two problems: (i) CEP with bounds on the amount of capacity that can be installed on each link, and (ii) CEP with unsplittable flows. That is, the demand between each origin destination pair must flow on a single path.

We study the Survivable Capacity Expansion Problem (SCEP) in Chapter 4. For this problem, we are required to add capacity on the links of the network so that all the demands can be routed even if a link in the network fails. We show that this problem is APX-Hard implying that, unless $P = NP$, there is no Polynomial Time Approximation Scheme for solving it. We then present two constant factor approximation algorithms for this problem, both using the Decentralized Routing Scheme that we presented in Chapter 3.

In Chapter 5, we examine capacity planning with time sharing. We present several models for the variability of peak origin-destination demands in a national long-distance network. We establish necessary and sufficient conditions for a feasible solution to the problem of finding capacities that can support demands at any point in time, and show that in its general form, this problem is NP hard. We identify and

develop algorithms for polynomially solvable special cases of this problem. We propose a heuristic solution procedure that solves a master problem iteratively by adding more violated cuts at each stage. When the set of possible demands is a polyhedron, we show that in a network with n nodes, the solution procedure we propose is an n -factor approximation in the worst case.

In Chapter 6, we computationally evaluate the solution procedure we presented for capacity planning with time sharing. We tested the heuristic on real telecommunication networks as well as randomly generated networks. For the instances we tested, the heuristic was able to produce solutions within 10% of optimality. More importantly, we show that as compared to optimal solutions without time sharing, the heuristic solutions that share capacity over time can save as much as 50% of the total expansion costs on average for the problem instances we tested.

We present conclusions and some directions for future research in Chapter 7.

1.3 Preliminaries

We introduce some terminology that will allow us to describe as well as provide a context for the problems we address. Demands for voice traffic are measured in a few different units. A DS0 (Digital Signal 0) refers to a bandwidth of 64 kilobits per second, and is the capacity used to route a single voice call. Twenty four DS0 signals can be multiplexed and carried over a higher capacity (1.5 Megabits per second) trunk called a DS1. A DS3 refers to the capacity equivalent to 28 DS1s or 672 simultaneous voice calls. Higher capacities are referred to as Optical Carrier (OC) signals. An OC-N has capacity equivalent to N-DS3s.

Time Division Multiplexing (TDM) is the traditional technology used to route calls in telecommunication networks in which many signals share a single medium (of higher bandwidth) by taking turns in using the medium for short time slots. Voice over Internet Protocol (VoIP) can also be used to route voice calls over a telecommunication network. In this case, a voice stream is broken into data packets that are transported to the call destination using packet switching.

A switch is the hardware that routes calls. Switches can be programmed to decide the paths on which the calls between each origin destination pair will be routed. Switches might also contain an Add Drop Multiplexer (ADM) that allows more efficient use of capacity by combining calls (multiplexing) as well as other equipment. A trunk is the medium connecting two switches. For long distance networks, the trunk is an optical fiber. Generally, capacity installed on a trunk between two switches can be used to route calls in both directions as long as the total number of calls routed is within the capacity of the trunk. In the networks that we consider, switches are the nodes and the trunks are the edges.

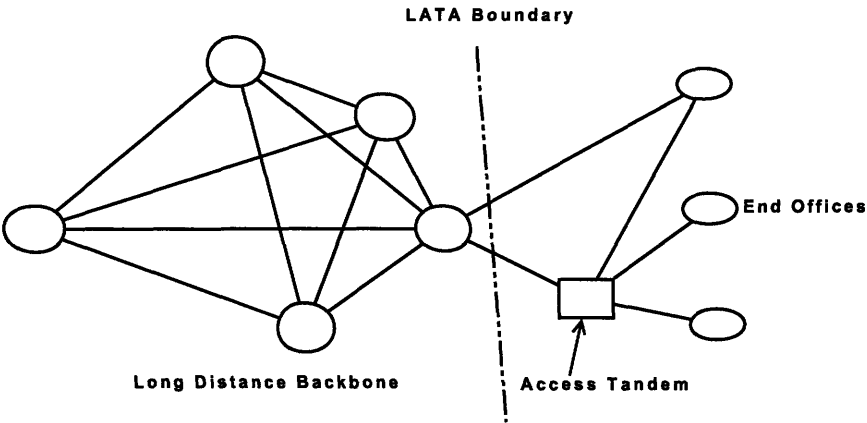


Figure 1-2: A telecommunication network

A telecommunication network (see Figure 1-2) consists of local access networks and a long distance backbone. The United States is divided geographically into areas called Local Access Transport Areas (LATA's). The part of the telecommunication network that is within a LATA is called the local access network. User telephone lines connect to End Offices which then connect to the long distance backbone through the local access network. Each switch in the long distance backbone handles traffic entering or leaving a large geographical area like a city. The long distance backbone carries a call only when the call's origin and destination are in regions covered by different long distance switches. This thesis addresses problems defined in the long distance backbone.

An algorithm A that produces a feasible solution to any instance of an minimiza-

tion problem P is a ρ -factor approximation algorithm for some $\rho > 1$ if the cost of the feasible solution produced by A is at most ρ times the optimal cost for every instance of P . The number ρ is also referred to as the *performance guarantee* of the algorithm. We can similarly define approximation algorithms for maximization problems.

An algorithm for P that runs in polynomial time and takes as input a positive number ϵ and produces a solution with cost at most $(1 + \epsilon)$ times the optimal cost for every positive ϵ is a Polynomial Time Approximation Scheme (PTAS). In addition, if the run time of the algorithm is bounded by a polynomial in $1/\epsilon$, the algorithm is a Fully Polynomial Time Approximation Scheme (FPTAS).

An optimization problem P belongs to the class APX if there it has a ρ -factor approximation algorithm for some constant $\rho > 1$. An L-reduction (or linear reduction) is a special type of polynomial reduction that preserves approximability. In particular, if an optimization problem P' belongs to APX, and P is L-reducible to P' , then P also belongs to APX. The toughest problems in the class APX are referred to as APX-Hard. A problem that is APX-Hard (for example, Three Dimensional Matching) does not admit a PTAS.

The Min-Knapsack Problem (MKP) is a minimization variant of the well known (Integer) Knapsack Problem in which we are given n items along with a knapsack of capacity W . Each item i has a size w_i and a penalty p_i if the item is not included in the knapsack. The objective is to choose a set of items that will fit in the knapsack so that the total penalty of the items left out is minimized. This problem allows an FPTAS, and many of the known FPTASs for the Knapsack Problem can be modified to obtain an FPTAS for the Min-Knapsack Problem. If we modify the FPTAS of Ibarra and Kim [29] (as presented in Korte and Vygen [32]), we obtain an FPTAS for the MKP with a running time of $O(n^2/\epsilon)$ for an instance with n items.

Another variant of the Knapsack Problem of interest to us is the Bounded Knapsack Problem in which multiple copies (up to a given item dependent limit) of an item could be included in the knapsack. Both this problem and its minimization variant admit an FPTAS.

We introduce relevant notation as needed. In general, though, throughout this

thesis we refer to an undirected edge between nodes i and j as $\langle i, j \rangle$ and we denote a link directed from i to j as (i, j) .

1.4 Literature review

1.4.1 Capacity Expansion Problem

Capacity Expansion Problems in telecommunication networks have been well studied in the literature. The survey paper by Luss [35] provides a summary of early work on these problems.

Researchers have studied several variants of the Capacity Expansion Problem. The expansion of the network could be carried out for a single period ([16, 12]) or for multiple periods ([50, 48, 18]). While in the single facility variant, capacity on the edges can be installed only in integral multiples of a base unit, in the multi-facility CEP ([16, 50, 48]), capacity can be installed in multiples of a few different units.

Bienstock and Günlük [16] consider the multifacility capacity expansion problem in a general telecommunication network. They present three classes of facet defining inequalities, and use them in a cutting-plane based solution procedure. Saniee [48] presents a dynamic programming algorithm for multi-facility capacity expansion of a single link.

Considerable attention has focused on capacity expansion on a special type of telecommunication networks called the local access networks (see survey papers by Gavish [23] and Balakrishnan et al. [9]). Local access networks contain a special sink node to which all the demand is destined, and the topology of the built network is restricted to be a tree. Shulman and Vachani [50] propose a decomposition approach for the local access network multi-period capacity expansion problem with two facilities. For the single period, multi-facility version of this problem, Balakrishnan, Magnanti and Wong [12] provide a Lagrangian relaxation based heuristic that solves subproblems using dynamic programming. They improve the heuristic by incorporating valid inequalities of the model into the dynamic program.

A related subject of interest with an extensive body of work is network loading. In the Network Loading Problem (NLP), also called the Buy-at-Bulk Problem, we seek a least cost way of installing facilities of different capacities on the links to meet given origin-destination demands. The NLP can be viewed as capacity expansion starting with a network with no initial installed capacity.

One major component of the literature on network loading considers polyhedral approaches ([15, 36, 37]): generating valid inequalities and using them in a cutting plane procedure. For the single period, two facility variant, Magnanti, Mirchandani and Vachani [37] present a cutting plane procedure that uses several families of valid inequalities, and compare this approach with a Lagrangian based approach. Bienstock et al. [15] address the single period, single facility variant, and propose two solution methods, one following a cutting plane procedure, and another that uses a cutting plane procedure followed by a branch and bound scheme.

Another approach in the NLP literature focusses on heuristics. For the single facility NLP, Mansour and Peleg [39] present an $O(\log n)$ -approximation algorithm using light weight distance preserving spanners. The approximation ratio of their heuristic is a constant for problems with Euclidean costs. Epstein [22] develops a family of heuristics that generates a tree solution for the NLP. He also presents a greedy heuristic with an $O(n)$ performance guarantee for networks with n nodes. Hassin, Ravi, and Salman [28], Salman et al. [47], and Gupta, Kumar, and Roughgarden [27] provide approximation algorithms for different versions of the NLP on networks with a single sink. Salman et al. [47] present a heuristic for a single sink multifacility Network Loading Problem with a guarantee that is logarithmic in the total demand. For the same problem, Gupta, Kumar, and Roughgarden [27] develop a 72.8-factor approximation algorithm. Hassin, Ravi, and Salman [28] describe an approximation algorithm for the single sink single facility NLP with a performance guarantee of 3.55. They initially use an approximate Steiner tree to route demands, and improve the cost by sending aggregated demands that are close to the facility capacity to the sink along a shortest path. When all the nodes in the graph have demands to the sink, their guarantee improves to 3.

1.4.2 Survivable Capacity Expansion Problem

Kennington, Olinick, and Spiride [31] provide a comprehensive survey of mathematical programming models for survivable capacity planning and expansion. Two kinds of failure protection have been studied in the literature: link restoration (or line restoration) and path restoration. Link restoration establishes alternative paths between the endpoints of the failed link to reroute demands using the link. Alternately, path restoration uses new paths between the origin and destination of demands when a primary path used by the demand has become unavailable due to some link failure. Kennington, Olinick, and Spiride present models for both link restoration and path restoration using either node-arc formulations or path flow based formulations.

Balakrishnan et al. [11] consider link restoration using a single facility type for installing spare (restoration) capacity. They provide polyhedral results, and devise a solution procedure using a cutting plane approach. They present computational results showing that the procedure performs well on three real world instances. For the multi-facility version of this problem, Balakrishnan et al. [10] present heuristics and solve instances with up to 50 nodes and 150 edges.

Veerasley, Venkatesan, and Shah [52] propose a path restoration heuristic that computes alternate paths for demands when any link fails. They consider one edge at a time, and heuristically assign alternate paths for all the demands that use this link.

Each of the previous two papers considers network restoration applied to a network with known current flows. A more general problem, which is the one we consider, would be to simultaneously choose flows ‘no fault’ routing (or primary flows) and restoration capacity. Lisser, Sarkissian, and Vial [34] propose a two stage solution procedure for this problem. The first stage computes the primary routing of the traffic assuming local rerouting. Fixing the primary routes, the second step identifies a global rerouting strategy.

Stoer and Dahl [51] study the integrated planning of both the base and the spare capacity under many failure scenarios (single node failure, single or multiple edge fail-

ures). Studying the projection of the formulation onto the capacity design variables, they derive valid inequalities and use them in a cutting plane procedure.

Bienstock and Muratore [17] study the problem in which a fixed fraction of demands must be transported even if a single edge or node fails. They present three different models for this problem, and describe several classes of facet defining inequalities for these models.

A related problem is survivable network design in which we seek to build a network with minimum cost so that there are at least a given number of alternate paths between node pairs. Grötschel, Monma and Stoer [26] present a review of polyhedral results for this problem. They also describe exact algorithms that use cutting-planes, and present computational results.

1.4.3 Network Planning with Capacity Sharing

The book by Clark [19] and Cisco's Voice Design and Implementation Guide [1] describe busy hour traffic estimation as well as traditional capacity planning in circuit switched telecommunication networks.

The Capacity Sharing Problem we address has been studied in the literature in a different context. The Robust Network Design Problem with demand uncertainty is mathematically equivalent to the Capacity Sharing Problem. We seek to design the network to support different demand vectors occurring at different points in time, whereas the Robust Network Design Problem seeks a capacity plan that supports any possible realization of a single uncertain demand vector. Due to this equivalence, we use some ideas and terminology from the Robust Optimization literature.

One such idea is that of adjustable robust optimization. Ben-Tal et al. [13] consider linear programs with uncertain parameters in which some variables, called adjustable variables, can be chosen after the uncertain parameters have been realized. They show that problem of choosing non-adjustable variables to minimize the worst case cost is NP-Hard in most cases. When the adjustable variables can be expressed as affine functions of the uncertain data, they show that these variables can be eliminated to obtain a robust linear programming problem (which can be written as a

linear program). In the context of capacity expansion, we can view the flow variables as adjustable variables as call routing can be done dynamically while capacity installation must be done in advance.

Researchers have studied robust versions of several network flow and design problems. Bertsimas and Sim [14] solve network flow problems with cost uncertainty (where each cost parameter is chosen independently from an interval) by solving a polynomial number of minimum cost problems. Kouvelis and Yu [33] consider the robust uncapacitated network design problem in which all parameters of the problem are uncertain. Assuming a finite number of scenarios, they propose a near optimal heuristic procedure using an adaptation of Benders decomposition.

Mudchanatongsuk, Ordonez and Liu [42] consider the robust network design problem in which both the cost and the demand parameters are uncertain and pursue an approach based on adjustable robust optimization by restricting the flow variables to be a linear function of the realized demand.

Perhaps the work closest to ours is the paper by Atamtürk and Zhang [7]. The authors consider the minimum cost network flow problem as well as the network design problem with uncertain demand in which some of the flow variables are adjustable. They consider two demand uncertainty sets: a cardinality set in which the number of demands that vary from the mean is limited, and a budget uncertainty set which restricts a weighted sum of the deviations from the mean of all demands. They provide a characterization for the optimal solution that includes an exponential number of cut constraints, and show that separating these constraints is NP-Hard. Computational results on a capacitated facility location problem indicates that their model could be a good alternative to a stochastic programming approach.

Chapter 2

Single Period Capacity Expansion

In this chapter, we study a model of capacity expansion in a hybrid long-distance telecommunication network that uses both TDM and VoIP technologies. Given the current capacities on the links of the network, the objective is to identify the least cost way to install additional capacity on the links to be able to route a deterministic demand forecast. The model we consider in simple is two respects: we optimize over a single period, and install capacity only in multiples of a single fixed integer. Both aspects of the model, however, are consistent with current practice. In the telecommunication industry, due to the low availability of capital for expansion causes network planners and the lack of adequate data to support dynamic analysis, network planners typically plan over a single period. In addition, rapid changes in the technology available to route calls restricts managers from planning for a long time horizon. Telecommunication planners typically plan capacities using one of these facility types: type T1 lines (the bandwidth to carry 24 simultaneous voice calls), type DS3 (672 calls) or type OC12 (8064 calls).

Our main result in this chapter is a constant factor approximation algorithm for this capacity expansion problem. We provide a model for this problem that incorporates the special requirements of the hybrid network, and use it to design algorithms with good performance guarantees. Our development is organized as follows: Section 2.1 formally defines the CEP in hybrid networks, and models it as the Hub-and-Spoke Capacity Expansion Problem (HSP). In Section 2.2, we prove that the HSP is

NP-Complete, and also show that, unless $P=NP$, there is no polynomial time approximation scheme for this problem. When the capacity of the facility type is large, we show, in Section 2.3, that the HSP is polynomially solvable by an efficient algorithm that we present. Section 2.4 presents a 2-factor approximation algorithm for HSP for situations with no initial capacity, and Section 2.5 presents a $(2 + \epsilon)$ -approximation algorithm for any $\epsilon > 0$ for the general case of HSP.

2.1 Modeling the Capacity Expansion Problem

A hybrid telecommunication network consists of a set of TDM switches and a set of softswitches. The softswitches are connected to each other by IP links (that carry voice as well as other IP traffic) and form a complete subnetwork, which we call the IP subnetwork. A TDM switch is connected to other TDM switches or to softswitches through a TDM *trunk*. Capacities on the TDM trunks or the IP links are given in number of simultaneous voice calls, in multiples of the given facility capacity. The current capacity on some of the TDM trunks might be zero.

Given a demand forecast, we seek to identify a capacity expansion plan and an associated routing for the demand that minimizes expansion cost.

The capacity expansion problem in a hybrid network has many unique characteristics. First, for several reasons, the optimization is carried out only for the TDM subnetwork - which includes all the switches of the network but only the TDM trunks. Organizationally, different planning engineers are usually responsible for the TDM and IP subnetworks, and they plan the network capacities independently with minimum exchange of information. Economically, the cost of capacity in the IP subnetwork is much cheaper, and the capacity is usually added in large amounts every several years. So, for the year in consideration, there typically is enough spare capacity in the IP subnetwork to handle any possible increase in demand. Finally, the IP subnetwork will also transport other types of traffic like data or video and hence capacity planning in that subnetwork should consider more than just the voice demand forecast.

Second, a TDM switch cannot be used as an intermediate node in the call route of any origin-destination demand. Since telecommunication companies want their long distance networks to move away from TDM towards a fully VoIP network, they would like to keep the expansion of their TDM network to a minimum. Using a TDM switch as an intermediate switch requires an undesired increase in TDM subnetwork capacity. Also, the ports, which are entry and exit points for demand, at TDM switches are more costly than those at softswitches. So, it is quite likely that even if TDM switches could be used as intermediate switches, an optimal solution would not use any.

In addition, we make a few assumptions, without loss of generality, for this problem:

Assumption 1. *The demand forecast is undirected.*

If the demands were directed, we simply add the demands in both directions.

Assumption 2. *The IP links have infinite capacity.*

Assumption 3. *Each TDM switch is connected to at least one softswitch.*

Since demand between two TDM switches cannot be routed through any intermediate TDM switch, if a TDM switch is not connected to any softswitch, then its entire traffic must be routed directly to all of its destination TDM switches, and we can eliminate the TDM switch from the network. Therefore, we assume each TDM switch is connected to at least one softswitch.

For convenience, we also complete the network as follows: If it contains no link between two switches, we create a link and set the initial capacity of this link to be zero, and set the cost of a facility on this link to be infinity. For the rest of this chapter we work with a complete graph, keeping in mind that the subgraph induced by links with non-zero initial capacity or finite facility cost satisfies assumptions 2 and 3.

For notational simplicity, a parameter or a decision variable with subscripts ij (for example, c_{ij}) corresponds to the undirected edge $\langle i, j \rangle$ between i and j . Therefore, unless explicitly stated otherwise, c_{ij} and c_{ji} refer to the same parameter.

An instance of the Capacity Expansion Problem in hybrid networks is given by a set \mathcal{T} of TDM switches and a set \mathcal{S} of softswitches, along with parameters c_{ij} and u_{ij} that denote, respectively, the cost of a facility (or the hardware that increases the link's capacity) and the initial capacity as measured by the number of simultaneous calls on the trunk between switches i and j . If both i and j are softswitches, then c_{ij} is zero and u_{ij} is infinity. The capacity of a single facility is C simultaneous calls. The demand forecast for which the network has to be designed is given by $\{d_{ij} : i, j \in \mathcal{T} \cup \mathcal{S}\}$. We assume all demands are integral. Let n and m denote the number of TDM switches and the number of softswitches.

Since demand from one softswitch to another can be routed within the IP subnetwork for free, we can ignore demands between softswitches. A demand between a TDM switch and a softswitch can be routed to any softswitch, and subsequently routed for free within the IP subnetwork from this softswitch to the destination. Therefore, we can merge all the softswitches in the set \mathcal{S} into a single switch called the *hub*.

The merger creates parallel links between TDM switches and the hub. Since in any optimal solution to the capacity expansion problem, we will add facilities only to the link out of a TDM switch with the cheapest facility cost, we could replace the parallel links with a single link with initial capacity equal to the total initial capacities on all the links connecting the TDM switch to any softswitch. The cost of a facility on this link is the minimum of the costs of facilities on all links between the TDM switch and any softswitch.

We now have a network (see Figure 2-1) with a central *hub* and trunks connecting the hub to each of the TDM switches. Note that our network also includes direct trunks between TDM switches. We refer to this network also as a hub-and-spoke network. We call the trunks between TDM switches *direct* links, and the trunks to the hub *radial* links. Therefore solving the hybrid network capacity expansion problem is equivalent to solving the following problem which we call the Hub-and-Spoke Capacity Expansion Problem (HSP): Given a hub-and-spoke network with nodes (\mathcal{T}, hub) , with costs c_{ij} , initial capacities u_{ij} , demands d_{ij} between the nodes

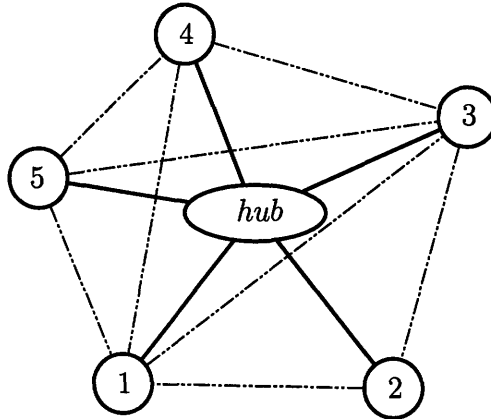


Figure 2-1: A hub-and-spoke network with direct links

and with facility capacity C , find a routing and the corresponding number of facilities to add on the trunks that minimizes the installation cost.

In this network, we can route all the demand between any TDM switch and a softswitch on the radial link, and adjust the initial capacity of that link accordingly. If necessary, we will install additional capacity on the radial links. The remaining demand is between pairs of TDM switches, and each of them can be routed in one of two ways: on the direct link between the two switches, or through the hub using two radial links. The HSP is essentially the problem of deciding, for each switch pair, how much of their demand travels on the direct link, and how much through the hub.

Since a direct link can be used to route only the demand between its end points, we first use the initial capacities on the direct links to route as much of this demand as possible. This brings either the demand or the initial capacity between these end-points to zero. If the demand becomes zero, then the remaining initial capacity on the direct link is unusable. Therefore, we can assume, without loss of generality, that the initial capacities of all direct links are zero in the given hub-and-spoke network. We number the switches in the hub-and-spoke network (other than the hub) arbitrarily.

The rest of this chapter paper deals with the Hub-and-Spoke Capacity Expansion Problem resulting from these modeling/preprocessing steps. We will establish the following results:

1. The Capacity Expansion Problem (CEP) in a hybrid network, in its general

form, is NP-Hard, and even APX-Complete.

2. For the uncapacitated version of the problem (where the facility capacity C is large), we present a fast polynomial time algorithm.
3. When there is no initial capacity on all the links of the network, we present a 2-approximation algorithm for the CEP
4. For the most general version of the CEP in a hybrid network, we present a $(2 + \epsilon)$ -approximation algorithm.

2.2 Complexity Results

Even though we are dealing with the capacity expansion problem on a fairly simple network topology, the HSP is NP-Hard. We show that this is true even for very restrictive special cases of the HSP. We provide a reduction from the weakly NP-Hard number partition problem to prove that the HSP is NP-Hard even when the network has a single source. As is well-known, the general single-source capacitated network design problem (which is defined for an arbitrary network, as opposed to a hub-and-spoke network for the HSP) is NP-Hard (see [47]), and our result shows that this is true even when the network structure is very simple.

Theorem 2.1. *The HSP is NP-Hard even when all initial capacities are zero and when all the demands originated from a single node.*

Proof. We polynomially reduce the well-known NP-Complete problem PARTITION to our problem. An instance of the PARTITION problem consists of n positive integers a_1, a_2, \dots, a_n . Let $B = \frac{1}{2} \sum_{i=1}^n a_i$. The problem is to ascertain whether there is a subset of these integers that sum to B .

Without loss of generality we assume that, in the given instance of the PARTITION problem, $a_n > 1$. We now create an instance of HSP as shown in Figure 2-2. In addition to the radial links shown in Figure 2-2, there are direct links between the node 0 and nodes $1, 2, \dots, n$. The facility capacity C is equal to B . The cost per

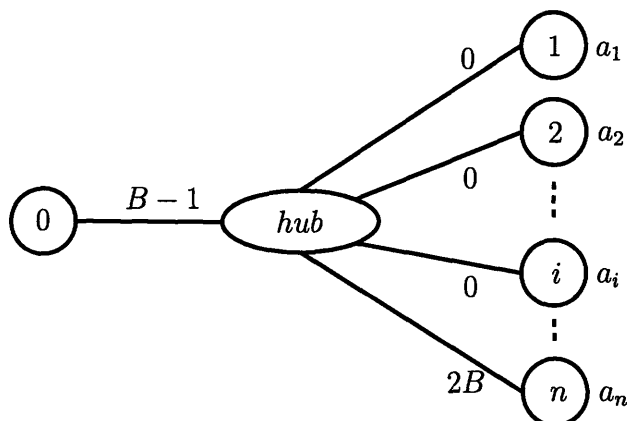


Figure 2-2: Reduction of PARTITION to SNLP

facility on the link between node 0 and the *hub* is $B - 1$, while it is $2B$ for the link $\langle n, hub \rangle$ and zero for all other links to the hub. The cost of each facility on the direct link (not shown in Figure 2-2) from node 0 to node i is a_i and the demands are:

$$d_{ij} = \begin{cases} a_j & \text{if } i = 0 \text{ and } j \in T, \\ 0 & \text{otherwise.} \end{cases} \quad (2.1)$$

We now show that the given instance of PARTITION is a YES instance if and only if the HSP has a feasible solution with cost not greater than $2B - 1$. Let $\mathcal{A} \subset \{1, 2, \dots, n\}$ and $\mathcal{A}' = \{1, 2, \dots, n\} \setminus \mathcal{A}$ with $\sum_{i \in \mathcal{A}} a_i = \sum_{j \in \mathcal{A}'} a_j$. Also, let $a_n \in \mathcal{A}'$. We route d_{0i} through the hub if $i \in \mathcal{A}$ and on the direct link otherwise, yielding a cost of $2B - 1$.

Conversely, if the HSP has a feasible solution with cost not greater than $2B - 1$, the cost structure implies that the number of facilities on $\langle 0, hub \rangle$ is at most one. But sending all the demand on direct links would cost $\sum_{i=1}^n c_{0i} = 2B > 2B - 1$, so the network must have exactly one facility on the link $\langle 0, hub \rangle$. Set

$$\mathcal{A}' = \{i | d_{0i} \text{ is routed directly in the given solution}\}$$

and $\mathcal{A} = \{1, 2, \dots, n\} \setminus \mathcal{A}'$. The capacity constraint on link $\langle 0, hub \rangle$ implies that

$$\sum_{i \in \mathcal{A}} d_{0i} = \sum_{i \in \mathcal{A}} a_i \leq B. \quad (2.2)$$

But since the total cost is at most $2B - 1$,

$$\sum_{i \in \mathcal{A}'} c_{0i} = \sum_{i \in \mathcal{A}'} a_i \leq (2B - 1) - (B - 1) = B. \quad (2.3)$$

Since $\sum_{i \in \mathcal{A}} a_i + \sum_{i \in \mathcal{A}'} a_i = 2B$, the inequalities (2.2) and (2.3) together imply that

$$\sum_{i \in \mathcal{A}} a_i = \sum_{i \in \mathcal{A}'} a_i = B,$$

that is, $\{\mathcal{A}, \mathcal{A}'\}$ is a partition.

In the HSP instance we created, all the demands originate at the node 0. So the HSP is NP-Hard even if all demands originate from a single source. \square

Without the restriction that all demands originate from the same switch, the problem becomes demonstrably tougher. One can show the following stronger result, by an L-Reduction from the APX-Hard 3-dimensional matching (3DM) problem. A proof of this result, due to Orlin [44], is given in Appendix A.

Theorem 2.2. *(Orlin [44]) The HSP is APX-Complete even when all the initial capacities are zero.*

This result implies that unless $P = NP$, no polynomial time algorithm would give solutions with cost arbitrarily close to the optimal cost. Therefore the best we can do is to develop approximation algorithms with constant factor performance guarantees. Before presenting approximation algorithms for the single period CEP (which reduces to HSP), we describe a special case that is polynomially solvable.

2.3 The Uncapacitated Problem

In the special case of the HSP when the facility capacity C is sufficiently large with respect to the given demands d , that is, for every TDM switch i of the network, $\sum_{j \in \mathcal{T} \setminus \{i\}} d_{ij} \leq C$, we never have to install more than one facility on any link. Suppose further that the initial capacities on the links of the network are all zero. We show that this special case is polynomially solvable by providing an integer programming model whose constraint matrix is totally-unimodular. We also present a fast primal dual algorithm for this special case.

Let $D = \{\langle i, j \rangle \mid i, j \in \mathcal{T}, d_{ij} > 0\}$ be the set of all non-zero demands. Define 0-1 decision variables x_{ij} for all $\langle i, j \rangle$ in the set D which assume value 1 if a facility is installed on the direct link between switches i and j . Similarly, define 0-1 variables y_i that take value 1 when a facility is installed on the radial link from switch i . We can formulate the uncapacitated hub-and-spoke capacity expansion problem as follows:

$$\begin{aligned}
 \text{(u-HSP) Minimize} \quad & \sum_{i,j \in \mathcal{T}; i < j} c_{ij} \cdot x_{ij} + \sum_{i \in \mathcal{T}} c_i \cdot y_i \\
 & x_{ij} + y_i \geq 1 && \forall \langle i, j \rangle \in D && (2.4a) \\
 & x_{ij} + y_j \geq 1 && \forall \langle i, j \rangle \in D && (2.4b) \\
 & x_{ij} \in \{0, 1\} && \forall \langle i, j \rangle \in D && (2.4c) \\
 & y_i \in \{0, 1\} && \forall i \in \mathcal{T}. && (2.4d)
 \end{aligned}$$

In this formulation, x_{ij} and x_{ji} refer to the same decision variable. The constraints (2.4a) and (2.4b) ensure that if x_{ij} is zero, both y_i and y_j are one. Therefore, any solution (x, y) satisfying the constraints (2.4) is a feasible solution to the uncapacitated HSP.

Theorem 2.3. *The uncapacitated HSP is polynomially solvable.*

Proof. Consider the following polyhedron \mathcal{P} defined by the linear programming re-

laxation of u-HSP:

$$x_{ij} + y_i \geq 1 \quad \forall \langle i, j \rangle \in D \quad (2.5a)$$

$$x_{ij} + y_j \geq 1 \quad \forall \langle i, j \rangle \in D \quad (2.5b)$$

$$x_{ij} \geq 0 \quad \forall \langle i, j \rangle \in D \quad (2.5c)$$

$$y_i \geq 0 \quad \forall i \in \mathcal{T}. \quad (2.5d)$$

Let A^* be the coefficient matrix in the matrix representation of the system of inequalities defining the polyhedron \mathcal{P} . We will show that A^* is totally unimodular, proving that \mathcal{P} is integral (see, for example, [32]). First, since constraints (2.5c) and (2.5d) are bound constraints, we can ignore them; let A be the matrix obtained from A^* by deleting the rows corresponding to constraints (2.5c) and (2.5d).

The transpose, A' , of the matrix A has exactly two 1s in every column. It is easy to see that A' is the node-arc incidence matrix of the undirected bipartite graph G constructed as follows. For every decision variable in (2.5) (x_{ij} or y_i), the graph G contains a vertex. For every nonzero demand d_{ij} , it contains two edges $\{x_{ij}, y_i\}$ and $\{x_{ij}, y_j\}$. Clearly, the node sets $\{x_{ij} : i < j \in \mathcal{T}\}$ and $\{y_i : i \in \mathcal{T}\}$ provide a bipartition of G .

From Theorem 5.24 of Korte and Vygen [32], the incidence matrix of an undirected bipartite graph is totally unimodular. This implies that A' , and therefore A and also A^* , is totally unimodular. \square

This result shows that the uncapacitated HSP can be solved in polynomial time by solving the linear program (2.5). However, the algorithm for solving the linear program needs to find a corner point solution. We now present an alternative polynomial time algorithm that has a much better run time complexity bound.

2.3.1 An Efficient Primal-Dual Algorithm

Consider the dual linear program of (2.5):

$$\text{Maximize } \sum_{i < j \in \mathcal{T}} (p_{ij}^i + p_{ij}^j) \quad (2.6a)$$

$$p_{ij}^i + p_{ij}^j \leq c_{ij} \quad \forall \langle i, j \rangle \in D \quad (2.6b)$$

$$\sum_{j \in \mathcal{T}: d_{ij} > 0} p_{ij}^i \leq c_i \quad \forall i \in \mathcal{T} \quad (2.6c)$$

$$p_{ij}^i \geq 0 \quad \forall \langle i, j \rangle \in D \quad (2.6d)$$

$$p_{ij}^j \geq 0 \quad \forall \langle i, j \rangle \in D. \quad (2.6e)$$

The dual problem allows the following interpretation: Each pair $\langle i, j \rangle$ corresponds to a supplier with capacity c_{ij} , and each switch i corresponds to a customer with demand c_i . Supplier $\langle i, j \rangle$ can supply only customers i and j . We would like to clear as many items as possible in this market. This is a variant of the well-known transportation problem.

We describe a simple primal-dual algorithm (see Dantzig et al. [20]) that uses the unique structure of this problem, and hence is fast. We start with the dual feasible solution $\{p_{ij}^i = 0, p_{ij}^j = 0 \mid i < j \in \mathcal{T} \text{ with } d_{ij} > 0\}$, and a set U of “unlabeled variables” containing all variables. We increase all the unlabeled variables uniformly until some dual constraint becomes tight. If this constraint is of type (2.6b), we set the corresponding primal variable x_{ij} to 1. If, on the other hand, the tight constraint is of type (2.6c), we set the corresponding primal variable y_i to 1. We label all the variables that appear in the dual constraint that became tight, and remove them from U . We repeat this procedure until all the variables are labeled. We show that this algorithm solves the uncapacitated HSP.

Theorem 2.4. *The primal-dual algorithm described above correctly solves the uncapacitated HSP. If the network has n switches, the running time of the algorithm is $O(n^4)$.*

Proof. Since we never alter the value of any variable once a constraint containing it

becomes tight, the dual is feasible throughout the algorithm. Next, we claim that when the algorithm terminates, the primal solution obtained is feasible. All the dual variables are labeled at termination. In particular, for each $\langle i, j \rangle$ in D , both p_{ij}^i and p_{ij}^j are labeled. Consequently, either the constraint

$$p_{ij}^i + p_{ij}^j \leq c_{ij}$$

is tight (implying $x_{ij} = 1$) or both constraints

$$\sum_{k \in T: d_{ik} > 0} p_{ik}^i \leq c_i \quad \text{and} \quad \sum_{k \in T: d_{jk} > 0} p_{jk}^j \leq c_j$$

are tight (implying both y_i and y_j are equal to 1). Therefore the obtained primal solution is feasible. By construction, the primal and dual solutions satisfy complementary slackness. Therefore, both solutions are optimal.

If the problem has n switches, the dual program has $O(n^2)$ variables and $O(n^2)$ constraints. Since in each iteration at least one new constraint becomes tight, the algorithm requires at most $O(n^2)$ iterations, and each iteration can be performed in $O(n^2)$ time. So the overall time complexity of this algorithm is $O(n^4)$. \square

Therefore the HSP without initial capacities is solvable in polynomial time if the facility capacity is sufficiently large. We also know that if the facility capacity is 1, we can solve the HSP efficiently (it is optimal to send all demands along the shortest origin-destination path). But when the facility capacity is arbitrary, we have shown that it is unlikely that the HSP has a Polynomial Time Approximation Scheme. So, we seek approximation algorithms with constant performance guarantee.

2.4 Hub-and-Spoke Networks Without Initial Capacities

In this section, we allow the facility capacity to be arbitrary. However, we restrict the initial capacity on each link in the network to be zero. In Section 2.2, we showed

that even this restrictive version of HSP is APX-Complete. We provide a 2-factor approximation algorithm for this problem based on a simple rounding technique. This algorithm is asymptotically optimal for large demands.

2.4.1 A Formulation For HSP Without Initial Capacities

We express the point-to-point demands d_{ij} as $d_{ij} = n_{ij}C + r_{ij}$, with $0 \leq r_{ij} < C$. We refer to $n_{ij}C$ as the *integral* demand, and r_{ij} as the *residual* demand.

Lemma 2.5. *The HSP without initial capacities has an optimal solution in which all the integral demands are routed along a shortest path with respect to the facility costs.*

Proof. Since a direct link $\langle i, j \rangle$ can route only demand between the nodes i and j , if we install any facilities on the link $\langle i, j \rangle$, we would route as much of demand d_{ij} on this link as possible. Consequently, the HSP has an optimal solution in which the demand routed along every direct link $\langle i, j \rangle$ is either an integral multiple of the facility capacity C or is equal to d_{ij} .

In such an optimal solution, all integral demands are routed along a shortest path. If not, at least C units of demand d_{ij} are routed on a path that is not the shortest origin destination path. We can strictly improve the solution by removing a facility along this path and installing it on the shortest path. We could move the C units of demand to the shortest path, contradicting the optimality of the solution. \square

Lemma 2.5 implies that given a HSP with demands d_{ij} , we can route the integral demands along shortest origin-destination paths, and concern ourselves only with the residual demands. Since the total cost of network expansion is the sum of the expansion costs for the integral and residual demands, a k -approximation algorithm for the HSP with residual demands alone yields a k -approximation algorithm for the original problem. For the rest of this section, we assume that there are no integral demands, *i.e.*, $n_{ij} = 0$ for every origin-destination pair $\langle i, j \rangle$.

As a result, the problem always has an optimal solution with the demand d_{ij} for each node pair $\langle i, j \rangle$ routed on a single path. If we route any of the demand d_{ij}

directly, we will need to install one facility on the direct link $\langle i, j \rangle$, which would have a capacity $C > d_{ij}$ and so we could route all of the demand directly at no extra cost. So the HSP has an optimal solution with the demand for each pair $\langle i, j \rangle$ routed either directly or through the hub in its entirety.

The problem reduces to deciding for each pair $\langle i, j \rangle$ whether we should transport the demand d_{ij} directly or through the hub. For each node pair $\langle i, j \rangle$ we define variables

$$x_{ij} = \begin{cases} 1 & \text{if } d_{ij} \text{ is routed through the direct link,} \\ 0 & \text{if } d_{ij} \text{ is routed through the } \textit{hub}. \end{cases}$$

Note that the decision variable x_{ij} also denotes the number of facilities to be built on the link $\langle i, j \rangle$. For each node i , we let y_i denote the number of facilities to be built on the link between the node i and the hub, and c_i the cost of installing a facility on this link. We can now model the HSP without initial capacities as the following integer program:

$$\begin{aligned} \text{(HSPa)} \quad \text{Minimize} \quad & \sum_{i,j \in \mathcal{T}; i < j} c_{ij} \cdot x_{ij} + \sum_{i \in \mathcal{T}} c_i \cdot y_i \\ & \sum_{j \in \mathcal{T} \setminus \{i\}} d_{ij} \cdot (1 - x_{ij}) \leq C \cdot y_i & \forall i \in \mathcal{T} \quad (2.7a) \end{aligned}$$

$$x_{ij} \in \{0, 1\} \quad \forall i, j \in \mathcal{T} \quad (2.7b)$$

$$y_i \in \mathbb{Z}^+ \quad \forall i \in \mathcal{T}. \quad (2.7c)$$

If we route a demand d_{ij} directly, we must install exactly one facility on the direct link $\langle i, j \rangle$ incurring a cost c_{ij} . Therefore, the objective function correctly represents the cost of loading the network. Constraint (2.7a) ensures there is enough capacity on the radial links to accommodate all demand that is not routed directly.

2.4.2 A Constant Factor Approximation Algorithm

Recall that facilities on a link to the hub can carry different demands, but a facility on a direct link $\langle i, j \rangle$ can carry only the demand d_{ij} . So, if for a node pair $\langle i, j \rangle$, it is more expensive to install a facility on the direct link than to install a facility along the origin destination path through the hub, *i.e.*, if $c_{ij} > c_i + c_j$, then it is optimal to send the demand through the hub. The HSP without initial capacities is trivial if the costs satisfy $c_{ij} > c_i + c_j$ for every node pair $\langle i, j \rangle$. So we assume that $\rho = \max_{i,j \in \mathcal{T}} \frac{c_i + c_j}{c_{ij}} > 1$.

Consider an optimal solution to the HSP, and let D be the set of demands d_{ij} that are routed directly. If we now change the routes of all the demands in D to go through the hub, we increase the cost of the solution by at most a factor of ρ .

Observation 2.6. *A solution to the HSP without initial capacities that routes all the demands through the hub has cost at most ρ times the optimal cost.*

If for the given problem, ρ is very close to 1, then we could send all the demands through the hub and obtain a near optimal solution. For problems in which this is not the case, we now provide a constant factor approximation algorithm.

The main idea underlying the algorithm is very simple: we relax the constraint that the number of facilities on the links to the hub should be integer, solve the relaxed problem, and round up the solution. For this simple algorithm to give near-optimal solutions, a few preprocessing steps are necessary. We describe these first.

If for some node i the cost of a single facility on the link between the node i and the hub is more expensive than the cost of sending all the demand out of node i directly, then in any optimal solution, we will send the demand directly. The following lemma formalizes this observation. Let $\delta(\cdot)$ be an indicator function, that is, $\delta(z) = 0$ if $z = 0$ and $\delta(z) = 1$ if $z > 0$. Then the cost incurred by sending all demands out of node i directly is $\sum_{j \in \mathcal{T} \setminus \{i\}} c_{ij} \cdot \delta(d_{ij})$.

If in an instance of the HSP without initial capacities, $\sum_{j \in \mathcal{T} \setminus \{i\}} c_{ij} \cdot \delta(d_{ij}) < c_i$ for some node i in \mathcal{T} , then it is optimal to route all the demands out of node i directly.

Therefore, we could perform the following preprocessing step before solving the HSP. If for any node i in \mathcal{T} , $\sum_{j \in \mathcal{T} \setminus \{i\}} c_{ij} \cdot \delta(d_{ij}) < c_i$, route all demand out of node i directly, and eliminate node i from the graph. Repeat this step until every node i in the current graph satisfies the condition $\sum_{j \in \mathcal{T} \setminus \{i\}} c_{ij} \cdot \delta(d_{ij}) \geq c_i$. For a reason that will become clear later, we do something stronger. We perform the preprocessing step repeatedly for every node i that satisfies the condition $\sum_{j \in \mathcal{T} \setminus \{i\}} c_{ij} \cdot \delta(d_{ij}) < 2c_i$ until we obtain a graph with no such node.

In the integer programming formulation for the HSP without initial capacities, if we relax the integrality constraints on the y variables alone, we can solve the resulting problem easily. In this case, we can buy fractional capacity on the radial links, and therefore the cost of sending a demand through the hub is proportional to the capacity used (that is, the demand). So the optimal solution routes each demand d_{ij} as follows: if c_{ij} is less than $(c_i + c_j) \cdot \frac{d_{ij}}{C}$, route the demand directly; otherwise, route the demand through the hub. More formally, when we relax the integrality constraints on the y variables, in an optimal solution to the mathematical program HSPa, the constraints (2.7a) must be satisfied as an equality. This result allows us to eliminate y variables, and the problem reduces to:

$$\text{Minimize}_{x_{ij} \in \{0,1\}} \sum_{i,j \in \mathcal{T}; i < j} \left(c_{ij} - (c_i + c_j) \cdot \frac{d_{ij}}{C} \right) \cdot x_{ij}.$$

We set the value of x_{ij} to be 1 if its coefficient is negative, that is, if c_{ij} is less than $(c_i + c_j) \cdot \frac{d_{ij}}{C}$.

Given an optimal solution to this relaxation, we can round all the y variables up to the next integer to obtain a feasible solution to the HSP. We call this procedure *Relax And Round*. If we performed this procedure on a hub-and-spoke network with arbitrary costs, the cost of the solution it produces can be arbitrarily bad. However, after performing the pre-processing steps we described, we show that this procedure has a constant performance bound.

Lemma 2.7. *Suppose that for a given instance of the HSP without initial capacities,*

$\sum_{j \in \mathcal{T} \setminus \{i\}} c_{ij} \cdot \delta(d_{ij}) \geq 2c_i$ for every node i in \mathcal{T} . Then, the *Relax And Round* procedure yields a solution to the HSP with cost at most two times the optimal cost.

Proof. Let Z^{OPT} , Z^{RLX} , and Z^{RAR} denote the optimal cost, the optimal cost of the relaxed problem, and the cost of the solution produced by the *Relax And Round* procedure respectively. Obviously, $Z^{RLX} \leq Z^{OPT}$. Consider an arbitrary optimal solution (x^*, y^*) to the integer program HSPa. We can write

$$Z^{OPT} = \frac{1}{2} \left(\sum_{i \in \mathcal{T}} \left(\sum_{j \in \mathcal{T} \setminus \{i\}} c_{ij} \cdot x_{ij}^* + 2c_i \cdot y_i^* \right) \right). \quad (2.8)$$

In any feasible solution (x, y) to HSPa, for each node i in \mathcal{T} at least one facility is loaded on the link between node i and the *hub*, or all the demands out of node i are routed directly. In the former case, $c_i \cdot y_i \geq c_i$, while in the latter case,

$$\sum_{j \in \mathcal{T} \setminus \{i\}} c_{ij} \cdot x_{ij} = \sum_{j \in \mathcal{T} \setminus \{i\}} c_{ij} \cdot \delta(d_{ij}) \geq 2c_i.$$

So, in either case,

$$\left(\sum_{j \in \mathcal{T} \setminus \{i\}} c_{ij} \cdot x_{ij} + 2c_i \cdot y_i \right) \geq 2c_i. \quad (2.9)$$

Summing these inequalities for the feasible solution (x^*, y^*) over all nodes in \mathcal{T} and substituting in (2.8), we obtain

$$Z^{OPT} \geq \frac{1}{2} \left(\sum_{i \in \mathcal{T}} 2c_i \right) = \sum_{i \in \mathcal{T}} c_i. \quad (2.10)$$

Now, in the *Relax And Round* procedure, rounding the y variables to the next integer increases the cost by at most $\sum_{i \in \mathcal{T}} c_i$, implying that

$$Z^{RAR} \leq Z^{RELAX} + \sum_{i \in \mathcal{T}} c_i \leq Z^{OPT} + Z^{OPT} = 2 \cdot Z^{OPT}.$$

□

Given any instance, we route all *integral demands* along shortest paths, and re-

peatedly remove nodes that have a ‘high’ cost for a facility on the link to the hub. After these preprocessing steps, we perform the *Relax And Round* procedure. The complete algorithm is given below:

THE ROUNDING METHOD(G, c, C, d)

Input: Undirected hub-and-spoke network G , facility costs for links c_i and c_{ij} , facility capacity C , and demands between nodes d_{ij} .

Output: Feasible solution to the HSP on G with cost at most two times the optimal cost.

- (1) *Routing integral demands:*
- (2) **foreach** node pair $\langle i, j \rangle$
- (3) **if** $d_{ij} \geq C$
- (4) (a) Let $d_{ij} = k_{ij} \cdot C + r_{ij}$, with $r_{ij} < C$
- (5) (b) **if** $(c_i + c_j \leq c_{ij})$ **then** route $k_{ij} \cdot C$ units out of d_{ij} through the hub
- (6) **else** route it directly
- (7) (c) Remove the routed demand, that is, set $d_{ij} = d_{ij} - k_{ij} \cdot C$

- (9) *Network Pruning:*
- (10) **while** \exists node i s.t. $\sum_{j \in \mathcal{T} \setminus \{i\}} c_{ij} \cdot d_{ij} < 2c_i$
- (11) (a) Route all demand out of i directly
- (12) (b) Remove node i from the network

- (14) *Relax And Round:*
- (15) **foreach** demand d_{ij}
- (16) **if** $((c_i + c_j) \frac{d_{ij}}{C} > c_{ij})$ **then** route d_{ij} directly
- (17) **else** route d_{ij} through the hub

- (19) *Cost Calculation:*
- (20) Using the routing obtained calculate the number of facilities to be installed on each link, and the total network expansion cost

Theorem 2.8. *The Rounding Method is a 2-approximation algorithm for the HSP without initial capacities. For a network with n switches, the running time of the method is $O(n^3)$.*

Proof. Since routing integral demands, the relax and round procedure and calculating the cost each requires $O(n^2)$ computations, and the network pruning procedure requires $O(n^3)$ steps (in a naive implementation), the algorithm has a running time of $O(n^3)$.

As routing integral demands maintains optimality, we assume that all demands are less than the facility capacity C . Let $1, 2, 3, \dots, k$ be the nodes that were eliminated by the network pruning procedure, and let G^i be the hub-and-spoke network after eliminating nodes $1, 2, \dots, i$. Let $Z^{RAR}(G^i)$ and $Z^{OPT}(G^i)$ be the cost of solution produced by the *Relax and Round* procedure and the optimal cost of the HSP restricted to the graph G^i respectively.

For any feasible solution (x, y) to the HSP with cost $Z(x, y)$, we can write

$$\begin{aligned} Z(x, y) &= \sum_{i \leq k} \left(\sum_{j \in G^i} c_{ij} x_{ij} + c_i y_i \right) + \sum_{i \in G^k} \left(\frac{1}{2} \sum_{j \in G^k \setminus \{i\}} c_{ij} x_{ij} + c_i y_i \right) \\ &\geq \sum_{i \leq k} \min \left(\sum_{j \in G^i} c_{ij}, c_i \right) + Z^{OPT}(G^k). \end{aligned} \quad (2.11)$$

The expression (2.11) provides a lower bound, Z^{LB} , on the value of any feasible solution to the given HSP. The network pruning procedure eliminates nodes i with $\sum_{j \in G^i} c_{ij} \leq 2c_i$, implying that

$$\sum_{j \in G^i} c_{ij} \leq 2 \cdot \min \left(\sum_{j \in G^i} c_{ij}, c_i \right) \quad 0 \leq i \leq k. \quad (2.12)$$

The cost, Z^{RM} , of the solution produced by the Rounding Method is

$$\begin{aligned} Z^{RM} &= \sum_{i \leq k} \sum_{j \in G^i} c_{ij} + Z^{RAR}(G^k) \\ &\leq \sum_{i \leq k} 2 \cdot \min \left(\sum_{j \in G^i} c_{ij}, c_i \right) + 2 \cdot Z^{OPT}(G^k) \\ &= 2 \cdot Z^{LB}. \end{aligned} \quad (2.13)$$

We conclude that the Rounding Method always produces a solution with at most twice the cost of an optimal solution. \square

2.4.3 Asymptotic Optimality of the Rounding Method

The rounding method always produces a solution to the HSP with cost at most twice that of an optimal solution. A factor of two might not be very attractive from a practical perspective. However, if the demands are large compared to the facility capacity C , this algorithm produces solutions with a performance guarantee much better than 2. The following result specifies a worst case bound that depends on the minimum demand to capacity ratio.

Theorem 2.9. *Let $k^* = \min_{i,j \in \mathcal{T}} \lfloor \frac{d_{ij}}{C} \rfloor$. The rounding method is an approximation algorithm to the HSP without initial capacities with a performance guarantee $(k^* + 2)/(k^* + 1)$.*

Proof. Let $k_{ij} \cdot C$ and r_{ij} denote the integral and residual demands respectively. Let Z^r be the optimal cost with demands r_{ij} . And let Z^C be the optimal cost when all demands are equal to C . Since the optimal costs are monotonically nondecreasing in the demand, $Z^r \leq Z^C$.

Let Z^* , Z^{RM} and Z^{ID} denote the optimal cost of the given HSP instance, the cost of the solution produced by the rounding method and the cost of optimally routing the integral demands respectively. Since for every node pair $\langle i, j \rangle$, $D_{ij} \geq k^* \cdot C$, $Z^{ID} \geq k^* \cdot Z^C$.

$$\begin{aligned}
 \frac{Z^{RM}}{Z^*} &= \frac{(Z^{ID} + Z^r) + (Z^{RM} - Z^{ID} - Z^r)}{Z^{ID} + Z^r} \\
 &= 1 + \frac{(Z^{RM} - Z^{ID} - Z^r)}{Z^{ID} + Z^r} \\
 &\leq 1 + \frac{(Z^{RM} - Z^{ID} - Z^r)}{k^* \cdot Z^C + Z^r} \\
 &\leq 1 + \frac{(Z^{RM} - Z^{ID} - Z^r)}{k^* \cdot Z^r + Z^r} \\
 &= 1 + \frac{1}{(k^* + 1)} \cdot \frac{(Z^{RM} - Z^{ID})}{Z^r} - \frac{Z^r}{(k^* + 1) \cdot Z^r}
 \end{aligned}$$

$$\begin{aligned} &\leq 1 + \frac{2}{(k^* + 1)} - \frac{1}{(k^* + 1)} \\ &= \frac{(k^* + 2)}{(k^* + 1)} \end{aligned}$$

□

The final inequality uses the result in Theorem 2.8 and when $k^* = 0$, we obtain the performance guarantee of two as established in that result. As k^* becomes large, the performance guarantee becomes better, reaching 1 asymptotically as k^* approaches infinity.

2.5 An Approximation Algorithm For The General Case

We now allow the initial capacities on the *radial* links to be nonzero. We first consider the case when these initial capacities are at most C . For capacity expansion problems in which we start from an empty initial network, after performing the modeling steps outlined in Section 2.1 we obtain a hub-and-spoke network in which each radial link has an initial capacity of at most C . We provide a lower bound for this problem using a decomposition scheme that involves solving a subproblem for each switch in the network. We model the subproblem as a generalization of the minimization version of the knapsack problem. With the help of this lower bound, we provide a $(2 + \epsilon)$ -approximation algorithm for any $\epsilon > 0$ for this problem. We then extend this algorithm, with a slight increase in computational complexity, to situations when the initial capacities can be arbitrary .

2.5.1 HSP With Small Initial Capacities

In this version of the HSP, we are given a hub-and-spoke network with a set \mathcal{T} of nodes, another (special) node called the *hub*, facility capacity C , facility installation costs c_{ij} and c_i for links $\langle i, j \rangle$ and links $\langle i, hub \rangle$. Each radial link has an initial

capacity u_i which is at most the facility capacity C . The problem is to route the given (undirected) demands d_{ij} to minimize the total cost of facility installation.

Since the initial capacity available along any path is less than C , routing C units of demand along any path requires installation of one facility along the path. So, retracing the proof for Lemma 2.5 for this case, we have the following lemma for the HSP with small initial capacities:

Lemma 2.10. *The HSP with small initial capacities has an optimal solution in which all the integral demands are routed along a shortest path with respect to the facility costs.*

Given an instance of the HSP with small initial capacities, we could route the integral demands along shortest paths. So, for the remainder of this section, we assume, without loss of generality, that the given demands are less than the facility capacity C . As in the case with no initial capacities, the problem reduces to deciding, for each node pair $\langle i, j \rangle$, whether to route the demand d_{ij} directly or through the hub. Therefore, we can write the following integer program for the HSP with small initial capacities:

$$\begin{aligned}
 \text{(HSPb) Minimize} \quad & \sum_{i,j \in \mathcal{T}, i < j} c_{ij} \cdot x_{ij} + \sum_{i \in \mathcal{T}} c_i \cdot y_i \\
 & \sum_{j \in \mathcal{T} \setminus \{i\}} d_{ij} \cdot (1 - x_{ij}) \leq u_i + C \cdot y_i \quad \forall i \in \mathcal{T}
 \end{aligned} \tag{2.14a}$$

$$x_{ij} \in \{0, 1\} \quad \forall i, j \in \mathcal{T} \tag{2.14b}$$

$$y_i \in \mathbb{Z}^+. \tag{2.14c}$$

We now provide a method to calculate a lower bound for the HSP. For each switch i in the hub-and-spoke network, consider the following (Expandable Knapsack)

optimization problem:

$$\begin{aligned}
 \text{(EKP-}i\text{)} \quad \text{Minimize} \quad & \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij} + c_i \cdot y_i \\
 & \sum_{j \in \mathcal{T} \setminus \{i\}} d_{ij} \cdot (1 - x_{ij}) \leq u_i + C \cdot y_i
 \end{aligned} \tag{2.15a}$$

$$x_{ij} \in \{0, 1\} \quad \forall j \in \mathcal{T} \setminus \{i\} \tag{2.15b}$$

$$y_i \in \mathbb{Z}^+. \tag{2.15c}$$

We interpret the problem EKP- i as a generalization of the 0 – 1 min-knapsack problem (MKP) (see, for example, [30], [40]). In the MKP, we are given a knapsack with size S , items with specified sizes, and penalties if the item is not included in the knapsack. The objective is to identify the set of items to be included in the knapsack in a manner that minimizes the total penalty of the items not included. In EKP- i , each demand $\langle i, j \rangle$ is an item with size d_{ij} and penalty $c_{ij}/2$. The decision variables x_{ij} indicate whether the item $\langle i, j \rangle$ has been included in ($x_{ij} = 0$) or excluded from ($x_{ij} = 1$) the knapsack. In this problem, the size of the problem is not fixed but can be expanded from u_i in integral multiples of C by paying a cost c_i . Therefore, EKP- i is a generalization of the MKP that allows expansion of the size of the knapsack and the objective is to minimize the cost of expansion and the penalty for unselected items.

If the optimal value y_i^* of the variable y_i is known, the problem reduces to an MKP: the size of the knapsack is $u_i + C y_i^*$, and the penalty of not including an item $\langle i, j \rangle$ in the knapsack is $c_{ij}/2$. The MKP is known to be NP-Hard, but, like the maximization version of the knapsack problem, there is a polynomial time algorithm that will find a $(1 + \epsilon)$ -approximate solution for any given $\epsilon > 0$. In fact, it is possible to modify almost all well-known fully polynomial time approximation schemes (FPTAS) for the maximization version of the Knapsack Problem to produce an FPTAS for the MKP (see, for example, Gens and Levner [24]). If we modify the fully polynomial time approximation scheme of Ibarra and Kim [29] for the Knapsack Problem (as presented in Korte and Vygen [32]), which, for a problem with n items, will solve the

MKP in $O(n^2/\epsilon)$ steps.

To solve the EKP- i , we can solve a sequence of MKPs, one for each possible value of the variable y_i . Since we assume that the given demands d_{ij} are at most C , the total demand out of any node is at most nC . Consequently, in EKP- i , $y_i \in \{0, \dots, n\}$, $\forall i \in \mathcal{T}$. Therefore, the algorithm in which we (approximately) solve a sequence of at most $(n + 1)$ MKPs is an FPTAS for EKP- i .

Let Z^i be the optimal cost of EKP- i . We will now show that $Z^{LB} = \sum_{i \in \mathcal{T}} Z^i$ is a lower bound on the optimal cost Z^* of the HSP with small initial capacities.

Theorem 2.11. Z^{LB} is a lower bound to the optimal cost of the HSP with small initial capacities.

Proof. Let (x^*, y^*) be an optimal solution to the HSP, *i.e.*, x_{ij}^* and y_i^* denote the number of facilities to be installed on the direct link $\langle i, j \rangle$ and the radial link from node i . Note that $(y_i^*, \{x_{ij}^* : j \in \mathcal{T} \setminus \{i\}\})$ is a feasible solution to EKP- i . Therefore,

$$Z^i \leq \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij}^* + c_i \cdot y_i^*$$

implying that

$$\begin{aligned} Z^{LB} &= \sum_{i \in \mathcal{T}} Z^i \\ &\leq \sum_{i \in \mathcal{T}} \left(\sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij}^* + c_i \cdot y_i^* \right) \\ &= \sum_{i \in \mathcal{T}} c_i \cdot y_i^* + \sum_{i, j \in \mathcal{T}; i < j} c_{ij} \cdot x_{ij}^* \\ &= Z^*. \end{aligned}$$

□

Suppose we are given (approximate) optimal solutions to the EKP- i problems for each node i in \mathcal{T} . We can create a feasible solution to the HSP as follows: If EKP- i and EKP- j both select an item (node pair) $\langle i, j \rangle$ in the knapsack, we route

the demand d_{ij} through the hub. We route all the other demands directly. This procedure, which we call the Decentralized Routing Algorithm, produces a feasible solution to the HSP. We show that the DRA solution has a constant performance guarantee.

Theorem 2.12. *The Decentralized Routing Algorithm for the Hub-and-Spoke Capacity Expansion Problem with small initial capacities is a $(2 + \epsilon)$ -factor approximation algorithm for any $\epsilon > 0$. For a network with n switches, the running time of the algorithm is $O(n^4/\epsilon)$.*

Proof. Consider how demand $\langle i, j \rangle$ is routed in the solution to the subproblems at switch i and j . Note that if the item is not selected in both solutions, the cost c_{ij} of routing d_{ij} directly is included in Z^{LB} . Now consider the set I of all items $\langle i, j \rangle$ that were included in the knapsack by either EKP- i or EKP- j but not both. Routing these demands directly increases the cost of the solution by at most $\sum_{\langle i, j \rangle \in I} c_{ij}/2$. However, since either EKP- i or EKP- j already paid the penalty for not including $\langle i, j \rangle$ in the knapsack, we have $Z^{LB} \geq \sum_{\langle i, j \rangle \in I} c_{ij}/2$, implying that the cost of the feasible solution to the HSP is at most $2Z^{LB}$. If, however, we start from $(1 + \epsilon/2)$ -approximate solutions for each of the EKP- i problems, we obtain a solution to the HSP in polynomial time with cost at most $(2 + \epsilon)Z^*$. Therefore, the procedure we have just outlined, which we call the Decentralized Routing Algorithm, is a $(2 + \epsilon)$ -approximation scheme for the HSP with small initial capacities.

For each subproblem, the algorithm solves at most $n + 1$ min-knapsack problems. Therefore, it solves a total of $O(n^2)$ min-knapsack problems with a total running time of $O(n^4/\epsilon)$. Obtaining a feasible solution from the solutions of the subproblems can be done with a run time that is linear in the number of demands, or $O(n^2)$. \square

2.5.2 Arbitrary Initial Capacities

We now consider the Hub-and-Spoke Capacity Expansion Problem when the initial capacities on the radial links are arbitrary. We note that if we apply the initial processing steps outlined in Section 2.1, the most general form of the CEP in hybrid

networks reduces to an HSP in which the radial links can have arbitrary initial capacities. Therefore, an (approximation) algorithm HSP with arbitrary initial capacities yields an (approximation) algorithm for the CEP in a hybrid network.

It is no longer optimal to route the integral demands along shortest paths as in the cases when the initial capacities were either absent or less than the facility capacity C . However, we show that the idea from Section 2.5.1 in which we obtained a lower bound by decomposing the problem into subproblems for each switch could still be used to obtain a constant factor approximation algorithm.

We duplicate each undirected demand d_{ij} into two directed demands D_{ij} and D_{ji} in opposite directions each with the same magnitude as d_{ij} . The demands D_{ij} can be either routed on the direct link between i and j , or can be sent to the hub (we do not send it to the destination switch in this case). The idea is to route the two demands D_{ij} and D_{ji} separately, ensuring that equal amounts of both is sent on the direct link between switches i and j . Such a routing yields a feasible solution to the underlying HSP. We provide an integer programming formulation for the HSP based on this idea.

Let the decision variable f_{ij}^d and f_{ij}^h denote the amount of demand D_{ij} that is routed directly and to the hub. Let y_{ij} and y_i be the number of facilities to be installed on the direct link between switches i and j , and the radial link out of switch i . Consider the following integer program:

$$\text{(HSPc) Minimize } \sum_{i \in \mathcal{T}} (c_i \cdot y_i + \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot y_{ij})$$

$$f_{ij}^d + f_{ij}^h = D_{ij}, \quad \forall i, j \in \mathcal{T} \quad (2.16a)$$

$$\sum_{j \in \mathcal{T} \setminus \{i\}} f_{ij}^h \leq u_i + C \cdot y_i \quad \forall i \in \mathcal{T} \quad (2.16b)$$

$$f_{ij}^d \leq C \cdot y_{ij}, \quad \forall i, j \in \mathcal{T} \quad (2.16c)$$

$$y_{ij} = y_{ji}, \quad \forall i < j \in \mathcal{T} \quad (2.16d)$$

All variables are integer.

The constraints (2.16a) ensure that all of the demand is sent either directly or to the hub. Constraints (2.16b) and (2.16c) size the facilities on the radial and direct links. Note that y_{ij} and y_{ji} are two different variables denoting the number of facilities on the direct link for the routing of D_{ij} and D_{ji} . Finally, constraints (2.16d) forces the number of facilities installed on the direct link from either end to be the same.

Proposition 2.13. *The integer program HSPc solves the Hub-And-Spoke Capacity Expansion Problem with arbitrary initial capacities.*

Proof. Let (y', f') be an feasible solution to (2.16). We can modify the solution, without changing the y variables, to ensure that as much of each demand flows on the direct link as possible. The new solution (y', f) has the same cost, and satisfies either $f_{ij}^d = y'_{ij}C$ or $f_{ij}^d = D_{ij}$ for every i, j . Since $y'_{ij} = y'_{ji}$, we have $f_{ij}^d = f_{ji}^d$ for every pair of directed demands D_{ij} and D_{ji} , implying that the part of demands D_{ij} and D_{ji} that go to the hub is also equal. If we use y' to install facilities on the links, and route f_{ij}^d units of the undirected demand d_{ij} on the direct link, we obtain a feasible solution for the HSP.

The two directed demands D_{ij} and D_{ji} each pay half the cost for the facilities on the direct link between switches i and j . Also, all the demands out of switch i pay for the facilities on the radial link out of switch i . Thus the objective function correctly accounts for the cost of added facilities.

Therefore, for each feasible solution of the integer program (2.16), we can obtain a feasible solution to the given HSP with the same cost. The optimal solution to (2.16) will give us the optimal solution to the HSP. \square

We now relax the constraints (2.16d) to obtain a lower bound. In the absence of constraint (2.16d), the problem decomposes into several subproblems, one for each switch. We call the subproblem at each switch the Local Routing Problem (LRP). For switch i the LRP is given by:

$$\text{(LRP-i) Minimize } c_i \cdot y_i + \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot y_{ij}$$

$$f_{ij}^d + f_{ij}^h = D_{ij}, \quad \forall j \in \mathcal{T} \setminus \{i\} \quad (2.17a)$$

$$\sum_{j \in \mathcal{T} \setminus \{i\}} f_{ij}^h \leq u_i + C \cdot y_i \quad (2.17b)$$

$$f_{ij}^d \leq C \cdot y_{ij}, \quad \forall j \in \mathcal{T} \setminus \{i\} \quad (2.17c)$$

All variables are integer.

Lemma 2.14. *LRP-i has an optimal solution in which for every demand D_{ij} , the amount routed on the direct link is either D_{ij} or $k \cdot C$, for some integer k .*

Proof. We use the argument used in the proof of Proposition 2.13. Given an optimal solution (y^*, f^*) to LRP-i, we can increase every demand f_{ij}^d until it either equals $y_i^* C$ or D_{ij} . Since we did not change y^* , the cost remains same, implying that the new solution, which satisfies the property stated in the lemma, is an optimal solution. \square

Lemma 2.14 implies that we can convert every demand $D_{ij} = k_{ij}C + r_{ij}$ into k_{ij} items of size C and one more item of size r_{ij} . If we know the optimal number of facilities, y_i^* , to add to the radial link from switch i , then the problem reduces to solving a min-knapsack problem with knapsack size $Cy_i^* + u_i$, and items obtained from the demands as described above. The penalty of not including an item from demand D_{ij} is $c_{ij}/2$. This problem can be solved using the $(1 + \epsilon)$ -approximation scheme of Gens and Levner [24]. There are two issues that remain: First, we do not know y_i^* . Second, when we create items from demands, we end up with a pseudopolynomial number of items.

To address the first issue, we observe the following: If an item has a penalty $c_{ij}/2$ that is at most the cost, c_i , of expanding the knapsack, this item will be included in the knapsack in an optimal solution. We can include all such items in the knapsack a priori (by increasing the size of the knapsack if necessary), and adjust the initial knapsack size u_i accordingly. Therefore, we can assume without loss of generality

that all items have a penalty lower than the cost of expanding the knapsack.

Lemma 2.15. *For the LRP- i problem in which the penalties are smaller than the knapsack extension cost, let y_i^* be the optimal number of times the size of the knapsack is increased. Then $y_i^* \in \{0, 1, 2, \dots, n\}$.*

Proof. We observe that if in an optimal solution to the LRP- i , $y_i > 0$, then all items of size C are not included in the knapsack. If this is not true, one such item is in the knapsack. We can remove this item from the knapsack incurring the penalty, and reduce y_i by 1. Doing so will reduce the cost since the penalty is lower than the knapsack extension cost c_i .

When y_i is greater than zero, only the residual demands r_{ij} can be included in the knapsack. The total residual demands is bounded by nC and, therefore, we never have to consider expanding the knapsack more than n times. \square

To avoid creating a pseudopolynomial number of items, we use a technique that has been applied to the Bounded Knapsack Problem (see for example [40]). Let $D_{ij} = k_{ij}C + r_{ij}$ and let t be an integer for which $2^{t-1} < k_{ij} \leq 2^t$. We split the demand D_{ij} into items of size $C, 2C, 4C, \dots, 2^{t-1}C, (k_{ij} + 1 - 2^t)C$ and r_{ij} . Let \hat{D} be the largest demand between any two switches. Then the total number of items in the LRP for any switch is bounded by $\hat{n} = n \log(\lceil \frac{\hat{D}}{C} \rceil)$.

So to solve LRP- i we can solve at most $n + 1$ min-knapsack problems, with knapsack sizes $u_i, u_i + C, \dots, u_i + nC$. Lemma 2.15 implies that one of these min-knapsack problems will have at most \hat{n} items, and the others will have n items. If we use the PTAS of Gens and Levner for the MKP, we can compute an ϵ -approximate solution to the LRP in $O((n^3 + \hat{n}^2)/\epsilon)$ time. Computing the ϵ -approximate lower bound, which involves solving the LRP for each switch, would then require $O(n(n^3 + \hat{n}^2)/\epsilon)$ time. From solutions to the LRPs for all the switches, we can obtain a feasible solution (\tilde{y}, \tilde{f}) to the HSP:

$$\tilde{y}_{ij} = \max(y_{ij}, y_{ji}) \text{ and } \tilde{f}_{ij}^d = \max(f_{ij}^d, f_{ji}^d) \quad \forall i, j \in \mathcal{T}.$$

Since this procedure is similar in overall structure to the one we described for HSP with small initial capacities, we refer to this algorithm also as the Decentralized Routing Algorithm (DRA).

Theorem 2.16. *The Decentralized Routing Algorithm produces a feasible solution to the HSP whose cost is at most $(2 + \epsilon)$ times the optimal cost.*

Proof. Let Z^* , Z^{LB} , and Z be the cost of the optimal solution, the lower bound, and the feasible solution produced by the DRA respectively. Let (y, f) be the solution corresponding to the lower bound. Then,

$$\begin{aligned} Z^{\text{LB}} &= \sum_{i \in T} (c_i \cdot y_i + \sum_{j \in T \setminus \{i\}} \frac{c_{ij}}{2} \cdot y_{ij}) \\ &\geq \sum_{i, j \in T: i > j} \frac{c_{ij}}{2} \cdot (y_{ij} + y_{ji}) \\ &\geq \sum_{i, j \in T: i > j} \frac{c_{ij}}{2} \cdot \max(y_{ij}, y_{ji}) \end{aligned}$$

Therefore,

$$\begin{aligned} Z &= Z^{\text{LB}} + \sum_{i, j \in T: i > j} \frac{c_{ij}}{2} \cdot (\max(y_{ij}, y_{ji}) - \min(y_{ij}, y_{ji})) \\ &\leq Z^{\text{LB}} + \sum_{i, j \in T: i > j} \frac{c_{ij}}{2} \cdot \max(y_{ij}, y_{ji}) \\ &\leq 2Z^{\text{LB}} \\ &\leq 2Z^*. \end{aligned}$$

□

2.5.3 A Tight Example

As shown by the example in Figure 2.5.3, there are instances of HSP for which the DRA produces solutions with cost twice that of the optimal solution. The per unit facility costs are indicated along the links. The initial capacities of all the links are zero. The facility capacity C is 1, and the demand between nodes 1 and 2 is also 1.

In the Decentralized Routing Algorithm for this example, we solve two subproblems, one each at switches 1 and 2.

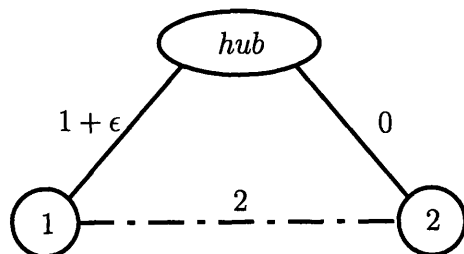


Figure 2-3: A bad example for the Decentralized Routing Algorithm

The subproblem at switch 1 seeks to send a total of a unit along the direct link or to the hub. Since the cost of sending the unit along the direct link is $1 = (\frac{1}{2}) \cdot 2$, it is optimal to send the entire unit along the direct link. On the other hand, for the subproblem at switch 2, it is optimal to route the demand to the hub at cost 0. In order to obtain a feasible solution to the HSP, the Decentralized Routing Algorithm will install a facility on the direct link. The cost of this solution is 2. However, the optimal solution is to route the demand through the hub at a cost of $1 + \epsilon$, which gives a performance ratio of 2 as ϵ goes to zero.

Chapter 3

Extensions of Hybrid Network Capacity Expansion Problem

The $(2 + \epsilon)$ -approximation scheme for the Hub-and-Spoke Capacity Expansion Problem (HSP) described in the previous section decomposes the problem into several subproblems, one at each switch, to compute a lower bound. This approach is quite general, and can easily be applied even if certain additional practical constraints need to be satisfied when solving the Capacity Expansion Problem for a hybrid network. With the additional constraints, the structure of the subproblem obtained will be different from the one we obtained for the HSP. However, we show that in several situations, an α -approximation algorithm for the resulting subproblem will translate into a 2α -approximation algorithm for the given problem when we use the algorithmic scheme proposed in the previous section.

In particular, we show that the Decentralized Routing Scheme is quite general and can be used for a variety of capacity expansion problems on hybrid networks. In Section 3.1, we describe several applications of the DR scheme, and show that the DR scheme converts an approximation algorithm for the subproblem obtained by the decomposition into an approximation algorithm for the original problem. We then study extensions to the single period capacity expansion problem in a hybrid network, and use the DR scheme to develop approximation algorithms for these problems. In Section 3.2, we consider the CEP with link dependent bounds imposed on the number

of facilities that can be installed on any link. We then study (in Section 3.3.1) the CEP in which each demand must be routed on a single path. After developing a $(2 + \epsilon)$ -approximation algorithm for the single facility type version of this problem, we extend the algorithm to handle multiple facility types as well.

3.1 The Decentralized Routing Scheme

We proposed a decentralized routing scheme that decomposes the HSP into subproblems and obtains a feasible solution with at most twice the sum of the costs of the subproblems. In this Section, we show that this scheme, while applicable only to a star network, is still very general. It can be used with almost any cost structure that separates by link, and for both deterministic or stochastic demand. It can also be used in a multi-period setting. The quality of the solution obtained based on this scheme depends on how well the subproblems can be solved or approximated.

Consider a general Capacity Expansion Problem in a hybrid network. We assume that the cost is a nondecreasing function of capacity, and it is separable by links. That is, the total cost of capacity expansion is the sum of the costs for each link in the network. Let $c_{ij}(x)$ be the cost of purchasing x units of capacity on link $\langle i, j \rangle$. The problem can have additional routing constraints (for example, unsplittability requiring that each demand is routed along a single path), capacity constraints (limits on the amount of capacity that can be installed on edges), or even survivability constraints (for example, single link failure protection).

In the hybrid network we will be considering, in any solution, part of each demand is sent on the direct link between its end points. The rest of this demand is routed through a path that consists of a link between its origin and a softswitch, a sub-path within the IP subnetwork, and a final link between a softswitch and its destination. We assume that the IP subnetwork has excess capacity, and hence is free.

We will decompose the problem into subproblems at each switch. At switch i , for each demand originating or terminating at switch i , we need to decide how much to route on the direct link, and how much to route on each link out of i to the IP

subnetwork. This routing is subject to all the additional constraints in the original problem. That is, if the original problem had limits on the capacity that can be added on certain links, we enforce these constraints in the subproblem. Finally, we set the cost function on the direct links to be $c_{ij}(x)/2$, while leaving the cost function on the other links unaltered.

When we have a feasible solution to all the subproblems, we can combine them to obtain a feasible solution to the original problem. If for the demand d_{ij} , the subproblems at i and j have routed unequal amounts on the direct link, we set the amount of demand d_{ij} routed directly to be the larger of these two. We then use this routing to obtain a feasible set of capacities to the original problem. We refer to this general procedure as the Decentralized Routing (DR) scheme.

When we use the DR scheme on a problem, the subproblems for each node are identical in the sense that they are different instances of the same problem. Also, this new problem is significantly simpler than the original problem because it no longer contains interaction between different switches.

Lemma 3.1. *Let P be a capacity expansion problem defined on a hybrid network, and let P^i be the subproblem at node i obtained by the Decentralized Routing Scheme. If for some α , there is an α -factor approximation algorithm for P^i , then the problem P has a 2α -approximation algorithm.*

Proof. We show that the Decentralized Routing Algorithm for P that uses an α -approximation algorithm for the subproblems is a 2α -approximation algorithm. Let Z^i be the cost of an optimal solution to the subproblem P^i , and let Z^* be the optimal cost of P . We first show that the sum $\sum_{i \in \mathcal{T}} Z^i$ is a lower bound on the optimal cost Z^* . Let (x, y) be an optimal solution to P . By definition of the subproblem P^i , the solution (x, y) restricted to the decision variables in P^i (we denote this solution by $(x, y)|_i$) is feasible to P^i . Let \hat{Z}^i be the cost of the feasible solution $(x, y)|_i$ to P^i . Observe that

$$\sum_{i \in \mathcal{T}} \hat{Z}^i = Z^*.$$

But, for every switch i , $\hat{Z}^i \geq Z^i$. We conclude that

$$\sum_{i \in \mathcal{T}} Z^i \leq Z^*.$$

Now, let Z_α^i be the cost of the solution returned by an α -approximation algorithm for the problem P^i . Then,

$$\sum_{i \in \mathcal{T}} Z_\alpha^i \leq \sum_{i \in \mathcal{T}} \alpha \cdot Z^i \leq \alpha \cdot \sum_{i \in \mathcal{T}} Z^i \leq \alpha \cdot Z^*. \quad (3.1)$$

The DR scheme generates a feasible solution to the problem P by setting the number of facilities \hat{x}_{ij} on a direct link $\langle i, j \rangle$ to $\max(x_{ij}, x_{ji})$, with x_{ij} and x_{ji} obtained from the α -approximation algorithms to the problems P^i and P^j respectively. Note that

$$\begin{aligned} Z_\alpha^i &\geq \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij}, \quad \forall i \\ \Rightarrow \sum_{i \in \mathcal{T}} Z_\alpha^i &\geq \sum_{i \leq j \in \mathcal{T}} \frac{c_{ij}}{2} \cdot (x_{ij} + x_{ji}) \\ \Rightarrow \sum_{i \in \mathcal{T}} Z_\alpha^i &\geq \sum_{i \leq j \in \mathcal{T}} \frac{c_{ij}}{2} \cdot \max(x_{ij}, x_{ji}). \end{aligned} \quad (3.2)$$

Let Z be the cost of the feasible solution to P produced by the Decentralized Routing Scheme. Then,

$$\begin{aligned} Z &= \sum_{i \in \mathcal{T}} Z_\alpha^i + \sum_{i \leq j \in \mathcal{T}} \frac{c_{ij}}{2} \cdot (\max(x_{ij}, x_{ji}) - \min(x_{ij}, x_{ji})) \\ &\leq \sum_{i \in \mathcal{T}} Z_\alpha^i + \sum_{i \leq j \in \mathcal{T}} \frac{c_{ij}}{2} \cdot \max(x_{ij}, x_{ji}) \\ &\leq \sum_{i \in \mathcal{T}} Z_\alpha^i + \sum_{i \in \mathcal{T}} Z_\alpha^i && \text{(from inequality (3.2))} \\ &= 2 \cdot \sum_{i \in \mathcal{T}} Z_\alpha^i \\ &\leq 2\alpha \cdot Z^* && \text{(from inequality (3.1)).} \end{aligned}$$

Therefore, the Decentralized Routing Algorithm that uses an α -approximation algorithm to obtain feasible solutions to the subproblems is a 2α -approximation for the given problem P . \square

To obtain a feasible solution to the problem, we transfer some of the demand routed to the hub to a direct route. This might allow us to reduce the number of facilities. We can postprocess the solution to ensure that only the required number of facilities are installed on the radial links. In the proof above, we assumed that no such postprocessing is done but the Lemma is obviously valid even when it is done.

We infer from Lemma 3.1 that the usefulness of the DR scheme in designing an approximation algorithm for a problem depends on the approximability of the obtained sub-problem. In Sections 3.2, 3.3.1, and 3.3.2, we show that for some extensions of the Single Period Capacity Expansion Problem, it is easy to approximate the sub-problem when applying the DR scheme to within a small constant factor, implying a constant factor approximation algorithms for these extensions. In Chapter 4, we consider the CEP in a hybrid network with survivability requirements. For this problem, applying the DR scheme directly does not give us subproblems that are easy to approximate. However, the DR scheme can still be useful as we can solve closely related relaxations using the scheme, and design heuristics based on these relaxations. We develop two constant factor approximation algorithms for this problem, both using the DR scheme.

Finally, we note that the decentralized routing scheme can be applied even to a multi-period capacity expansion problem in a hybrid network. In this context, we are given a multi-period demand forecast, and we seek to identify a capacity expansion plan (number of facilities to install on each link during each time period) that will satisfy the forecasted demand for each period. We seek to minimize the total (discounted) cost of expansion.

For the multiperiod problem, the decentralized routing algorithm would do the following: After obtaining solutions to the switch subproblems, the number of facilities on a direct link in each time period would be set to the maximum of the number of facilities installed in the subproblem solutions of the endpoints of the direct link in

that period. It is easy to show that Lemma 3.1 is valid for the multiperiod problem.

3.2 Upper Bounds On The Number Of Facilities

The Capacity Expansion Problem we considered in Chapter 2 does not limit the number of facilities that can be installed on links. The solution produced by the algorithms we have proposed in Sections 2.4 and 2.5 could send considerable demand on a few links, which could cause a huge disruption of service if one or more of these links fail. For this reason, network planners to prefer to distribute the load on the network as evenly as possible without increasing the cost of expansion very much. One way to do this is to set limits on the number of new facilities that can be installed on each link.

Also, until now we have been assuming that capacity in the IP subnetwork is free. Though for a typical year, there might be enough spare capacity in the IP subnetwork to handle increase in demand, routing lots of demand through this subnetwork would eventually result in an increase in infrastructure cost to the company. Therefore it is desirable not to send too much demand to the IP subnetwork. This could also be ensured by limiting the number of facilities that can be installed on the links between legacy switches and softswitches.

So, consider a hybrid network capacity expansion problem $(G((S \cup I), E), u, c, C, d)$ with bounds $\{n_e : e \in E\}$ imposed upon the number of new facilities that can be installed on the links. We assume, as we did earlier, without loss of generality, that the initial capacities u_e on direct links (between two legacy switches) is zero. We show how to use the framework of the Decentralized Routing Algorithm to obtain a $(2 + \epsilon)$ -approximation scheme for this variant of the CEP.

3.2.1 Routing Demands To Softswitches

Demands between two softswitches will be routed entirely within the IP subnetwork and therefore contribute nothing to the cost. We now consider demands between

legacy switches and softswitches. Since capacity inside the IP subnetwork is free, we need to route these demands to some softswitch. For each legacy switch s , we route the demands between switch s and all softswitches as follows: Let D_s be the sum of all the demands between the switch s and any softswitch. Similarly, let u_s be the sum of all initial capacities on links between switch s and any softswitch. We route the demand D_s to use up as much of the capacity u_s as possible. If D_s exceeds u_s , we need to buy additional capacity, and we start with the link with the cheapest facility cost and move to more expensive ones as the links reach their bounds on number of facilities added. We adjust the total initial capacity between the switch s and the IP subnetwork appropriately.

As for the case with no bounds on the number of new facilities, we can shrink the IP subnetwork to a hub node, and remove duplicate links to the hub to obtain a star network. However, the cost function for facilities on the radial links is no longer linear; it is a piecewise linear convex function in which the breakpoints occur whenever we reach the upper bound for a link.

3.2.2 Bounds On The Number Of Facilities On Direct Links

We show that without loss of generality, we can assume no bounds on the number of new facilities on direct links (between two legacy switches). If there is a bound n_e on a direct link e between legacy switches s and t , and the demand, d_{st} between s and t is no greater than $n_e C$, then we can ignore the bound. On the other hand, if d_{st} exceeds $n_e C$, we have to route at least $\Delta_{st} = d_{st} - n_e C$ through the hub in any feasible solution. So we route Δ_{st} units of demand through the hub, and remove the bound on the link e . To route Δ_{st} through the hub, we need only increase the aggregated demands D_s and D_t by Δ_{st} before routing the demands to softswitches. This procedure can be applied for each direct demand, thereby eliminating bounds on all the direct links.

3.2.3 A Model For CEP With Upper Bounds

We assume no bounds on the number of facilities added on direct links. As we did for the HSP, we create two directed demands, D_{ij} and D_{ji} , for every pair i, j of TDM switches. Each of these demands is equal in magnitude to the given demand d_{ij} between the switches. For every softswitch l , the decision variable y_{il} denotes the number of facilities to be installed on the link between the TDM switch i and the softswitch l . If the network does not contain a link (i, l) to a softswitch l , we set n_{il} to be zero. When we carry out the preprocessing steps to route demands to softswitches, or to eliminate upper bounds on direct links, we might add as many facilities on some links (i, l) as we are allowed. For these links, we also set n_{il} to be zero. We now formulate the CEP with upper bounds as an integer program. All other decision variables in this model are as in formulation HSPc (Section 2.5.2).

$$\begin{aligned}
 \text{(B) Minimize} \quad & \sum_{i \in \mathcal{T}} \left(\sum_{l \in \mathcal{S}} c_{il} \cdot y_{il} + \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot y_{ij} \right) \\
 & f_{ij}^d + f_{ij}^h = D_{ij}, \quad \forall i, j \in \mathcal{T} \quad (3.3a) \\
 & \sum_{j \in \mathcal{T} \setminus \{i\}} f_{ij}^h \leq u_i + C \cdot \left(\sum_{l \in \mathcal{S}} y_{il} \right) \quad \forall i \in \mathcal{T} \quad (3.3b) \\
 & f_{ij}^d \leq C \cdot y_{ij}, \quad \forall i, j \in \mathcal{T} \quad (3.3c) \\
 & y_{il} \leq n_{il}, \quad \forall i \in \mathcal{T}, l \in \mathcal{S} \quad (3.3d) \\
 & y_{ij} = y_{ji}, \quad \forall i < j \in \mathcal{T} \quad (3.3e)
 \end{aligned}$$

All variables are integer.

We now apply the DR scheme to this problem: we relax the constraints (3.3e), to obtain a lower bounding problem. This problem decomposes into subproblems, one for each switch. The subproblem is similar to the LRP obtained when we decomposed the formulation HSPc (Section 2.5.2), except that the cost of the radial link out of each TDM switch is piecewise linear and convex. We continue to represent these piecewise linear costs by using the variables $\{y_{il} : l \in \mathcal{S}\}$ for each TDM switch i .

The subproblem for switch i , which we call B- i , is given by:

$$\text{(B-}i\text{)} \quad \text{Minimize} \quad \sum_{l \in \mathcal{S}} c_{il} \cdot y_{il} + \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot y_{ij}$$

$$f_{ij}^d + f_{ij}^h = D_{ij}, \quad \forall j \in \mathcal{T} \setminus \{i\} \quad (3.4a)$$

$$\sum_{j \in \mathcal{T} \setminus \{i\}} f_{ij}^h \leq u_i + C \cdot \left(\sum_{l \in \mathcal{S}} y_{il} \right) \quad \forall i \in \mathcal{T} \quad (3.4b)$$

$$f_{ij}^d \leq C \cdot y_{ij}, \quad \forall i, j \in \mathcal{T} \quad (3.4c)$$

$$y_{il} \leq n_{il}, \quad \forall l \in \mathcal{S} \quad (3.4d)$$

All variables are integer.

3.2.4 A Decentralized Routing Algorithm For CEP With Upper Bounds

We note that Lemma 2.14 is valid for the B- i . So for each demand $D_{ij} = k_{ij}C + r_{ij}$ we create k_{ij} items of size C and an item of size r_{ij} . We interpret B- i as an expandable minimum knapsack problem: The initial size of the knapsack is u_i , but it can be expanded, and the cost of expansion is piecewise linear and convex. Each item created out of demand D_{ij} has a penalty $c_{ij}/2$. The objective is to identify the size of the knapsack and the items to include in the knapsack to minimize the total of the knapsack expansion cost and the penalties of items not included in the knapsack.

For ease of exposition, we assume that the cost of facilities on direct links out of the node i are all distinct. All the results we establish under this assumption are valid even when multiple direct links have the same facility cost.

Lemma 3.2. *In an optimal solution to the B- i , the items of size C are included in the knapsack in descending order of their penalties.*

Proof. Let i and j be two items of size C , and assume the penalty of item i be greater than that of item j . If in any feasible solution, item j is included in the knapsack and item i is not, we could exchange the two. The resulting solution is feasible and has lower cost. So in an optimal solution, if item j is in the knapsack, then item j

must also be in the knapsack. □

Lemma 3.2 implies that in any optimal solution to B- i , we route the integral demands in descending order of the cost of installing a direct facility.

Property 3.3 (Link Tightness Property). *A solution to B- i is said to satisfy the link tightness property if it satisfies one of the following conditions:*

- (Radially tight) For every softswitch l , either $y_{il} = n_{il}$ or $y_{il} = 0$.
- (Directly tight) For every TDM switch j , $f_{ij}^d = 0, r_{ij}$ or D_{ij} .

We show that B- i has an optimal solution that satisfies the link tightness property. For each switch j , let U_j (resp., L_j) denote the set of switches l with direct link facility cost c_{il} greater (resp., lesser) than c_{ij} .

Theorem 3.4. *There is an optimal solution to the B- i that satisfies the link tightness property.*

Proof. Let (\hat{y}_i, \hat{f}_i) be an optimal solution to B- i that satisfies Lemma 2.14, but does not satisfy the link tightness property. Lemma 3.2 implies that some switch j in the solution (\hat{y}_i, \hat{f}_i) sends all of the integral demand to switches in U_j to the hub, and all of the integral demand to switches in L_j directly. So, for each switch l other than j , \hat{f}_{il} is either 0, r_{il} or D_{il} . Since we assumed that (\hat{y}_i, \hat{f}_i) does not satisfy the edge tightness property, we conclude that f_{ij} must be equal to pC for some integer $0 < p < k_{ij}$. We also conclude that $0 < \hat{y}_{il} < n_{il}$ for some softswitch l .

Since we can shift C units of demand D_{ij} from the direct link to the radial link (i, l) or vice versa, the optimality of (\hat{y}_i, \hat{f}_i) implies that $c_{ij}/2 = c_{il}$. We can now shift as much demand from the direct link (i, j) to the radial link (i, l) until we either have no demand being sent on the direct link, or the number of facilities on the radial link has reached its bound n_{il} . This solution has the same cost as (\hat{y}_i, \hat{f}_i) , and so is optimal. It also satisfies the link tightness property. □

To obtain an optimal solution to B- i , we evaluate all solutions that are either radially tight or directly tight, and select the best solution among them. Since facilities on the radial links will be added in ascending order of facility cost, we need to

consider at most m possible radially tight solutions. Also, for each of these cases, we know the number of facilities to be installed on each radial link. We can solve the B- i in this case by solving at most m minimum knapsack problems.

We now consider the directly tight solutions. Lemma 3.2 implies that in any such solution some TDM switch j satisfies the property that for all TDM switches k with $c_{ik} \leq c_{ij}$ the entire integral part of the demand D_{ik} is sent directly, and for all other demands, the entire integral part is sent through the hub. Now, only the residual demands are left to be routed. Since the number of additional facilities on the radial links in any solution is at most n , we solve at most $n+1$ minimum knapsack problems, varying the number of additional facilities on the radial links from 0 to n . Selecting the best solution among these would give us the directly tight solution. We also note that there are at most n possible directly tight solutions.

We can use the PTAS for the minimum knapsack problem as a subroutine in this procedure to obtain an ϵ -approximate solution to B- i . This, when used in the framework of the Decentralized Routing Algorithm, gives a $(2 + \epsilon)$ -approximation algorithm for the CEP with upper bounds.

3.3 Unsplittable Demands

We study a variant of the Hub-and-Spoke Network Capacity Expansion Problem (HSP) in which we are required to route every demand on a single path. That is, every demand in its entirety must be routed either directly or through the hub. Traditionally, calls in a telecommunication network were routed along a single ‘primary’ path for every origin-destination pair. When the primary path does not have sufficient capacity to route a call (that is, it is ‘blocked’), the system will use an overflow path. Some network routers do not support percentage routing which is required if we were to allow splitting of the demands along different paths. Also, having a single primary route for each demand eases implementation and troubleshooting in case of failures. Owing to these two factors, network planners prefer to use a single path for each demand in certain situations, motivating our study of the Unsplittable HSP.

We are given a hub-and-spoke network with a set \mathcal{T} of switches along with a *hub* switch, and demands d_{ij} between switches i and j in \mathcal{T} . We initially study the single facility version in which capacity can be installed on each link in multiples of the facility capacity C . The cost of installing a facility is c_{ij} on the direct link $\langle i, j \rangle$, and c_i on the radial link between the switch i and the *hub*. Later, we extend the algorithm to multiple facility types as well. Let x_{ij} be the indicator decision variable that takes the value 1 when the demand d_{ij} is routed directly. We formulate the Unsplittable HSP as the following integer program:

$$\begin{aligned}
(\mathbf{U}) \quad \text{Minimize} \quad & \sum_{i \in \mathcal{T}} c_i \cdot x_i + \sum_{i < j \in \mathcal{T}} c_{ij} \left\lceil \frac{d_{ij}}{C} \right\rceil \cdot x_{ij} \\
& \sum_{j \in \mathcal{T} \setminus \{i\}} d_{ij} \cdot (1 - x_{ij}) \leq u_i + C \cdot x_i & \forall i \in \mathcal{T} & \quad (3.5a) \\
& x_i \in \mathbb{Z}^+, & \forall i \in \mathcal{T} \\
& x_{ij} \in \{0, 1\}, & \forall i < j \in \mathcal{T}.
\end{aligned}$$

Constraint (3.5a) ensures that enough capacity is installed on radial links to support demands that are routed through the hub.

We design an approximation algorithm for this problem using the Decentralized Routing Scheme. We duplicate each demand d_{ij} into two directed demands D_{ij} and D_{ji} , and for each directed demand, we need to choose between routing the demand on the direct link or sending the demand to the hub. For a hub-and-spoke network with n nodes, this decomposes the problem into n subproblems. The difference between this subproblem and the one obtained from the splittable HSP (2.17) is that the demands must now be routed in their entirety along one of the two paths. Here is the subproblem obtained for node i :

$$\begin{aligned}
(\mathbf{U-}i) \quad \text{Minimize} \quad & c_i \cdot x_i + \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot \left\lceil \frac{d_{ij}}{C} \right\rceil \cdot x_{ij} \\
& \sum_{j \in \mathcal{T} \setminus \{i\}} D_{ij} \cdot (1 - x_{ij}) \leq u_i + C \cdot x_i & (3.6a)
\end{aligned}$$

$$\begin{aligned}
x_i &\in \mathbb{Z}^+ \\
x_{ij} &\in \{0, 1\}, & \forall j \in \mathcal{T} \setminus \{i\}.
\end{aligned}$$

We interpret this problem as a generalization of the min-knapsack problem. The demands are items with size equal to the magnitude of the demand. The radial link is the knapsack with capacity u_i . The penalty of not including an item in the knapsack (that is, routing the corresponding demand on the direct link) is $(c_{ij}/2) \cdot \lceil d_{ij}/C \rceil$. The size of the knapsack, however, can be increased in multiples of C , and the cost per expansion is c_i . We refer to this problem as the Expandable Min-Knapsack Problem.

Given this interpretation, we are interested in designing an approximation algorithm for the Expandable Min-Knapsack Problem. From Lemma 3.1, we know that an α -approximation algorithm for the problem U- i can be used to obtain a 2α -approximation algorithm for the unsplittable HSP. In Section 3.3.1, we develop an FPTAS for the Expandable Min-Knapsack Problem, implying a $(2+\epsilon)$ -approximation algorithm for the Unsplittable HSP for any positive ϵ .

3.3.1 The Expandable Min-Knapsack Problem

In the Expandable Min-Knapsack Problem, we are given a set of items $\{1, 2, \dots, n\}$ with sizes w_i and penalties p_i . We are also given a knapsack with initial size W that can be expanded in multiples of a given expansion size C for a cost c^K per expansion. The goal is to decide the size of the knapsack, and the items to include that will minimize the total of the expansion cost and the penalties for items not included. Let the decision variable x_i be 1 when the item is not included in the knapsack and 0 otherwise. Letting y be the number of times we expand the knapsack, we can write this problem as the following integer program:

$$\begin{aligned}
(\mathbf{E}) \quad \text{Minimize} \quad & c^K \cdot y + \sum_{1 \leq i \leq n} p_i \cdot x_i \\
& \sum_{1 \leq i \leq n} w_i \cdot (1 - x_i) \leq W + C \cdot y
\end{aligned} \tag{3.7a}$$

$$\begin{aligned}
y &\in \mathbb{Z}^+ \\
x_i &\in \{0.1\}, & \forall 1 \leq i \leq n.
\end{aligned}$$

We show that the problem does not become harder by allowing expansion of the knapsack.

Theorem 3.5. *The Expandable Min-Knapsack Problem is equivalent to the Min-Knapsack Problem. There is a Fully Polynomial Time Approximation Scheme for the Expandable Min-Knapsack Problem.*

Proof. Rewrite the constraint (3.7a) as follows:

$$C \cdot y + \sum_{1 \leq i \leq n} w_i \cdot x_i \geq \sum_{1 \leq i \leq n} w_i - W. \quad (3.8)$$

The constraint (3.8) allows us to interpret the knapsack expansions also as items. The problem is to choose a set of items of minimum ‘cost’ whose total size is at least $\sum_{1 \leq i \leq n} w_i - W$. This is exactly the min-knapsack problem, except that the variable y is allowed to assume integer values larger than 1. We create ‘expansion items’ of size $C, 2C, 4C, \dots, 2^t C$, with $(2^{t+1} - 1)C \geq \sum_{1 \leq i \leq n} w_i - W$. The cost of these items are $c_i, 2c_i, \dots, 2^t c_i$ respectively. We can then expand the knapsack to any allowable size by choosing the right set of expansion items. It is easy to verify that the number of items created is polynomially bounded.

We conclude that the Expandable Min-Knapsack Problem is as easy as the Min-Knapsack Problem. We can use the indicated transformation to convert any Expandable Min-Knapsack Problem to the Min-Knapsack Problem. Therefore, any FPTAS for the min-knapsack problem yields an FPTAS for the Expandable Min-Knapsack Problem. \square

3.3.2 Multiple Facility Types

When the demands are unsplittable, it is easy to extend the prior algorithm to be applicable to the HSP with multiple facility types. Now we would be able to in-

crease capacity in several different sizes. We are given k facility types with capacities C^1, C^2, \dots, C^k , and we can install one or more facilities of each type on every edge of the network. The cost c_e^ℓ of the facilities depends on the type ℓ and the edge e . Usually, the costs of the facility types exhibit economies of scale; that is, if $C^1 < C^2 < \dots < C^k$, $c_e^1/C^1 > c_e^2/C^2 > \dots > c_e^k/C^k$. We assume that k is a small number. In telecommunication applications, k could typically be 2 or 3.

Since the demands are unsplittable, if we decide to route a demand d_{ij} on the direct link (i, j) between switches i and j , we need to install facilities of different types on the direct link to minimize the total cost of facilities installed. Since k is a small number, it is easy to determine the optimal installation. Let the cost of this installation be c_{ij} . We can then formulate the problem as the following integer program:

$$\begin{aligned}
(\text{M}) \quad \text{Minimize} \quad & \sum_{i \in \mathcal{T}} \sum_{1 \leq \ell \leq k} c_i^\ell \cdot x_i^\ell + \sum_{i < j \in \mathcal{T}} c_{ij} \cdot x_{ij} \\
& \sum_{j \in \mathcal{T} \setminus \{i\}} d_{ij} \cdot (1 - x_{ij}) \leq u_i + \sum_{1 \leq \ell \leq k} C^\ell \cdot x_i^\ell & \forall i \in \mathcal{T} \\
& x_i^\ell \in \mathbb{Z}^+, & \forall i \in \mathcal{T}, \forall 1 \leq \ell \leq k \\
& x_{ij} \in \{0, 1\}, & \forall i < j \in \mathcal{T}.
\end{aligned}$$

The variable x_i^ℓ is the number of facilities of type ℓ installed on the radial link out of switch i , and x_{ij} assumes the value 1 if the demand d_{ij} is routed directly. We note that this problem is the same as **U**, except that facilities of more than one type can be installed on the radial links. As for the single facility case, we apply the decentralized routing scheme to obtain subproblems for each node of the network. Again, the subproblems differ from **U**- i only with respect to the facility types available to install capacity on the radial link.

Consider the subproblem for switch i . We can interpret the subproblem as an expandable knapsack problem, but in this case, the knapsack's capacity can be expanded using more than one facility type. However, this does not change the problem. For each facility type ℓ , we create items with size $C^\ell, 2C^\ell, \dots, 2^t C^\ell$, with

$(2^{t+1} - 1)C^\ell \geq \sum_{j \in T \setminus \{i\}} d_{ij} - u_i$. The cost of these items are $c_i^\ell, 2c_i^\ell, \dots, 2^t c_i^\ell$ respectively. We now need to solve a min-knapsack problem with these items as well as the demand items. Again, the total number of items is polynomially bounded. Therefore, using the FPTAS for the min-knapsack problem, we obtain an FPTAS for the subproblem, and thereby a $(2 + \epsilon)$ -approximation algorithm for the Multi-facility Unsplittable HSP.

Chapter 4

Capacity Expansion with Single Link Failure Protection

In the Survivable Capacity Expansion Problem (SCEP), we are given a hybrid telecommunication network with costs of facilities (of a given capacity) on the links, and a single year demand forecast. We seek to identify primary routes (or “no-fault” routes) for the demand, i.e., paths on which calls are routed under normal conditions, as well as secondary routes that can be used to send demand whenever a link in one of the primary paths fails. The objective of the SCEP is to identify primary and secondary routes for all origin destination pairs, and decide how many facilities to add to each of the links of the network so that all demand can be carried even when any link of the network fails, rendering the capacity of the link unavailable.

This chapter is organized as follows: We develop a compact integer programming formulation for the SCEP in Section 4.1, and show that the problem is APX-Hard. In Section 4.2, we introduce the Bounded Network Restoration (BNR) problem that is useful for developing approximation algorithms for the SCEP. We provide a polynomial time algorithm for the BNR. We enumerate four lower bounds for the SCEP in Section 4.3. Finally, in Section 4.4, we develop two constant factor approximation algorithms for the SCEP. The first algorithm has a performance guarantee of $(5 + \epsilon)$, and the second algorithm improves this ratio to $(4 + \epsilon)$.

4.1 Modeling the SCEP in a Hybrid Network

We make a few assumptions associated with a hybrid network: A TDM switch cannot be an intermediate switch in routing demands, and the IP subnetwork has a large amount of spare capacity. For the Survivable Capacity Expansion Problem, the latter assumption implies that protection against IP link failures is guaranteed within the IP subnetwork, i.e., whenever an IP link fails, the IP network has a large amount of excess capacity, including an alternate path between the endpoints of the failed link. Therefore, we can ignore failures of IP links.

Also, for the same reason, we can transport demand from any softswitch to any other softswitch for free. So, as we did for the CEP, we can merge the entire IP subnetwork to a single hub node. This possibly creates parallel links between TDM switches and the hub node. Since we would like to protect against single link failures, we are interested in scenarios in which at most one of these parallel links fails. If we replace the parallel radial links by single links as we did for the CEP, we will not be able to analyze these scenarios separately. Therefore we do not replace parallel links. Figure 4-1 shows the hybrid network after this preprocessing.

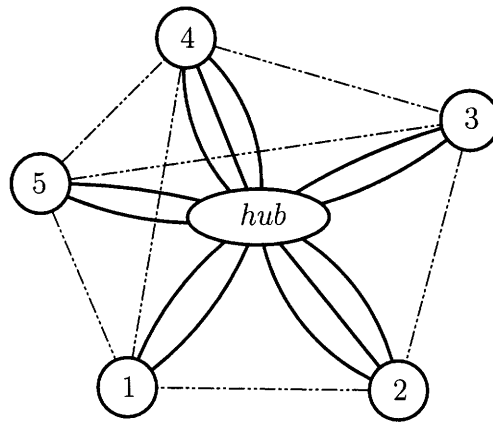


Figure 4-1: A star network with parallel radial links

We also assume, without loss of generality, that each TDM switch has at least two radial links to the hub. If there is demand between a TDM switch i and softswitches, feasibility (single link failure protection) dictates that there be at least two radial links incident to switch i . In the absence of any demand to softswitches, if a TDM

switch i had just one radial link, then the switch can be removed from the network. In this case, the radial link is used to reroute traffic if a direct link out of switch i fails. We install enough capacity on the radial link to ensure that we can reroute any of the direct demands out of switch i . We can then remove the switch i from the network.

We now provide a compact formulation for this problem. We introduce the following notation: d_{ij} is the demand between switches i and j , C is the capacity of a facility, and c_{ij} is the cost of a facility on the link between switches i and j . The initial capacity (in number of facilities) of the link between switches i and j is u_{ij} .

For each TDM switch i , we denote the set of radial links out of the switch i to the hub by $R(i)$. Let d_i be the total demand between the TDM switch i and any softswitch. TDM switches cannot be intermediate switches in routing, implying that the direct link $\langle i, j \rangle$ between TDM switches i and j is available to route only the demand d_{ij} . Therefore, any additional capacity to protect against link failures must be installed only on the radial links. Since any of the radial links can transport demand to the hub, we need to know only the total demand that is carried from a TDM switch to the hub to decide how many facilities to install on the radial links. Therefore, we define a decision variable f_{ij}^h as the total amount of demand d_{ij} routed through the hub. Let the variable f_{ij}^d denote the amount of demand d_{ij} sent on the direct link $\langle i, j \rangle$. Also, let x_{ij} and y_e be the number of new facilities on the direct link $\langle i, j \rangle$ and on the radial link e . f_i is the total demand that is transported on all the radial links out of TDM switch i . We formulate the SCEP as the following integer program:

$$\begin{aligned}
\text{(SCEP) Minimize} \quad & \sum_{i < j \in \mathcal{T}} c_{ij} \cdot x_{ij} + \sum_{i \in \mathcal{T}} \sum_{e \in R(i)} c_e \cdot y_e \\
& f_{ij}^d + f_{ij}^h = d_{ij}, & \forall i < j \in \mathcal{T} & \quad (4.1a) \\
& f_{ij}^d \leq C \cdot (u_{ij} + x_{ij}), & \forall i < j \in \mathcal{T} & \quad (4.1b) \\
& \sum_{j \in \mathcal{T}} f_{ij}^h + d_i \leq f_i, & \forall i \in \mathcal{T} & \quad (4.1c)
\end{aligned}$$

$$C \cdot \left(\sum_{\ell \in R(i)} (u_\ell + y_\ell) - (u_e + y_e) \right) \geq f_i, \quad \forall e \in R(i), \forall i \in \mathcal{T} \quad (4.1d)$$

$$C \cdot \sum_{\ell \in R(i)} (u_\ell + y_\ell) \geq f_i + f_{ij}^d, \quad \forall i, j \in \mathcal{T} \quad (4.1e)$$

$$y_e \geq 0, \text{ and } y_e \text{ integer} \quad \forall e \in R(i), i \in \mathcal{T}$$

$$f_{ij}^d, f_{ij}^h \geq 0, \quad \forall i < j \in \mathcal{T}$$

$$f_i \geq 0, \quad \forall i \in \mathcal{T}.$$

This formulation ensures that even if one of the links incident to any TDM switch i fails, the network has sufficient capacity to transport all the demands either to their destination or to the hub. The constraint (4.1d) guarantees that the available capacity on all the radial links is enough to carry the demand transported to the hub if one of the radial links fails. The constraint (4.1e) ensures that the network has enough spare capacity on the radial links to transport the extra demand when the traffic on a failed direct link is rerouted to go through the hub. It is clear that a solution to the integer program (4.1) satisfies the survivability requirement. We note that even when one radial link incident on *each* TDM switch fail at the same time, the solution will have enough available capacity to transport all demands.

Most models for survivability in telecommunication networks in the literature contain multiple copies of the demand routing constraints, one copy for each possible state (in our case, the failed link or the normal, no failure state) of the network (see [2, 3, 31, 41]). The number of variables and constraints in these models is quadratic in the number of links in the network, and since these models are multicommodity formulations, the size of the model is also proportional to the number of commodities. In comparison, the model we have proposed for the SCEP is compact: it grows linearly with the number of links in the network. For reasonably sized networks, the model in (4.1) could be solved as such using an IP solver.

To obtain primary and secondary call routes from a solution to the integer program (4.1), we do the following. For primary routes, we solve a feasible flow problem using origin-destination demands and the number of facilities available on each link

of the network. We could include an objective to reduce the splitting of demands on several paths. To identify secondary routes under each link failure, we fix the unaffected primary routes, and solve a feasible flow problem only for the demands that are affected by the failure. In a network with m links, we need to solve a total of $m + 1$ feasible flow problems to identify primary and secondary routes.

The SCEP is a computationally hard problem. A straightforward approximation preserving reduction from the HSP proves the following for the Survivable Capacity Expansion problem.

Theorem 4.1. *The Capacity Expansion Problem with single link failure protection in a hybrid network is APX-Hard.*

Proof. We provide an L-reduction from the Hub-and-Spoke Network Capacity Expansion Problem (HSP) which we showed to be APX-Hard in Chapter 2. In particular, we show that given an instance of the HSP, we can create an instance of the SCEP in polynomial time so that for every feasible solution to the instance of the HSP, there is a corresponding feasible solution to the created instance of the SCEP with the same cost.

Let the instance of the HSP be given by a hub-and-spoke network $G = (V, E)$ with hub node h , cost vector \mathbf{c} and demand vector \mathbf{d} . We create an instance of SCEP by adding to the network G zero cost parallel links to every radial link $\langle i, h \rangle$ in G . The new network G' is shown in Figure 4-2. The costs of all the other links are the same as in G . The demands between switches are the same as in the HSP.

Let the vector \mathbf{y} be the number of facilities installed on the edges of the network G in an arbitrary feasible solution to the HSP. To define a feasible solution for the SCEP, we need to specify the number of facilities on every link in G as well as the newly created radial links. We define a solution \mathbf{x} to the SCEP as follows: For every link $\langle i, j \rangle$ in G , including the original radial links, we set $x_{ij} = y_{ij}$. For every new zero cost radial arc $\langle i, h \rangle$, we set

$$x_i^o = \left\lceil \frac{1}{C} \max(u_i + C \cdot y_i, \max_{j \in S \setminus \{i\}} C \cdot y_{ij}) \right\rceil.$$

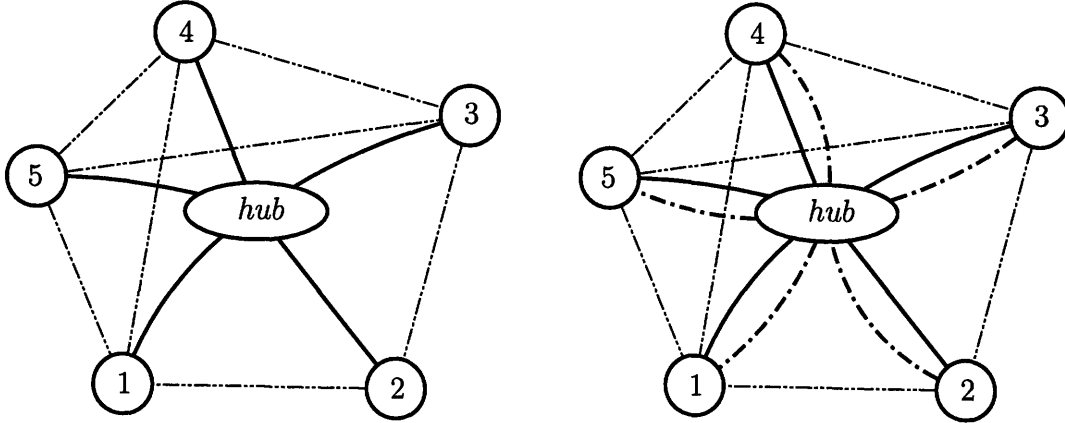


Figure 4-2: The L-Reduction for the SCEP

The cost of this solution is obviously the same as the cost of the solution \mathbf{y} to the HSP. When a direct link fails, the zero cost alternate route through the hub will have enough capacity to transport the demand. And when a radial link fails, the zero cost radial link can be used to route the flow instead, showing that \mathbf{x} is feasible for the SCEP.

Conversely, let $\mathbf{x} = (\mathbf{y}, \mathbf{z})$ be a feasible solution to the SCEP, with \mathbf{y} denoting the number of facilities on the links in G and \mathbf{x} the number of facilities on the zero cost radial links. We claim that \mathbf{y} is a feasible solution to the HSP. Since \mathbf{x} is feasible to the SCEP, we know that even when a zero cost radial link fails, there is enough capacity to route all demands. Mathematically, for every switch i in the network,

$$u_i + C \cdot y_i \geq \sum_{j \in S \setminus \{i\}} \min(d_{ij}, C \cdot y_{ij}).$$

Consequently, \mathbf{y} is a feasible solution to the HSP and, as in the previous case, it is easy to see that the cost of this solution is the same as the cost of \mathbf{x} .

Since we L-Reduced the APX-Hard HSP to the SCEP, we conclude that SCEP is APX Hard. \square

This result precludes the existence of a polynomial time approximation scheme for the SCEP.

4.2 Parallel Path Network Restoration Problem

We consider the parallel path network restoration problem (NR) defined by Magnanti and Wang [38] (also see [53]). In this problem, two nodes are connected by a number of parallel links (see Figure 4-3). For each link e , we are given a ‘demand’ d_e , i.e., the minimum number of facilities that must be present between the two nodes even if link e fails. Note that the demand is specified in number of facilities (alternatively, we can think of the facility capacity as being 1). The objective is to decide the number of facilities to install on the links so that the demand requirements are satisfied with minimum total installation cost.

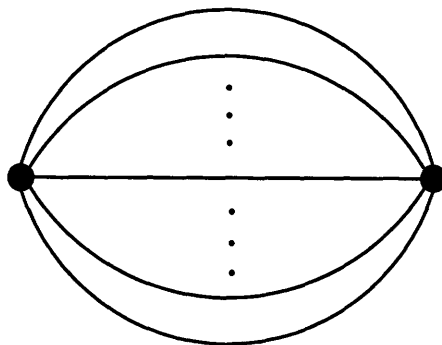


Figure 4-3: A two-node parallel edge network

Let c_e denote the cost of link e , and let E be the set of parallel links. The problem can be formulated as the following integer program:

$$\begin{aligned}
 \text{(NR) Minimize} \quad & \sum_{e \in E} c_e \cdot y_e \\
 & \sum_{f \in E} y_f - y_e \geq d_e, & \forall e \in E & \quad (4.2a) \\
 & y_e \geq 0, \text{ and } y_e \text{ integer} & \forall e \in E.
 \end{aligned}$$

The variable y_e denotes the number of facilities to be installed on link e . Magnanti and Wang provide a complete description of the convex hull for this problem. They also present a fast polynomial time algorithm for the problem. Bienstock and Mura-tore [17] consider a similar problem in which all the demands are the same, with an

additional constraint restricting the total number of facilities on all the link to be at least a pre-specified number. They present several classes of facet defining inequalities for this problem.

We seek to solve the NR in the presence of bounds on the number of facilities that can be installed in each of the links. This problem, which we call the bounded network restoration (BNR) problem, generalizes the problems studied by both Wang [53] and Bienstock and Muratore [17]. We are interested in the problem because it appears as a subproblem in our heuristics for the SCEP. Let n_e be the maximum number of facilities that can be installed on link e . We assume that with these values of n_e , the problem is feasible. A formulation for the Bounded Network Restoration problem is just the integer program NR along with the constraints $y_e \leq n_e$ for every link e .

However, for the purpose of developing an algorithm for the BNR, we define an additional variable Y , and let it to be the total number of facilities installed on all the links. This idea is similar to the one used by Magnanti and Wang [38] for the Network Restoration problem. We modify the integer program (4.2) to obtain the following formulation for the BNR problem:

$$\text{(BNR) Minimize } \sum_{e \in E} c_e \cdot y_e$$

$$y_e \leq Y - d_e, \quad \forall e \in E \quad (4.3a)$$

$$y_e \leq n_e, \quad \forall e \in E \quad (4.3b)$$

$$\sum_{e \in E} y_e = Y \quad (4.3c)$$

$$y_e \geq 0, \text{ and } y_e \text{ integer} \quad \forall e \in E$$

$$Y \geq 0, \text{ and } Y \text{ integer.}$$

In this formulation, Y is a variable. If we know the value of Y in an optimal solution, the number of facilities on each link can be easily obtained using the following greedy procedure. Sort the edges in ascending order of the facility costs c_e . In this order, we install $\min(Y - d_e, n_e)$ facilities on each edge e until we have installed a total of Y facilities. It is easy to check that this procedure produces a solution with minimum

cost when total number of facilities is Y . For a network with k parallel links, the greedy procedure requires $O(k)$ time.

The greedy procedure can also identify if the problem is infeasible for a particular value of Y . In this case, we will run out of edges before we can install a total of Y facilities. That is, $\sum_{e \in E} \min(Y - d_e, n_e) < Y$.

Now, let $v(Y)$ equal the optimal cost of the linear relaxation of the integer program (4.3) as a function of the total number Y of facilities installed. A simple argument (basic linear programming theory) implies that the function $v(Y)$ is piecewise linear and convex. Let Y^* be the value of the variable Y in an optimal solution to the linear relaxation of the integer program (4.3). We know that the integer program (4.3) has an optimal solution with the total number of facilities Y equal to either $\lfloor Y^* \rfloor$ or $\lceil Y^* \rceil$, and therefore we can check both and choose the solution with lower cost. Also, we note that when Y is integer, the greedy procedure gives an integer solution to the linear relaxation, that is, the y_e variables are all integer. We can use the greedy procedure after identifying an optimal Y to find the number of facilities y_e on each link e . This polynomial time algorithm for the problem is similar to the one Magnanti and Wang [38] present for the Network Restoration Problem without bounds. They also present a linear time combinatorial algorithm to solve the linear programming relaxation, which cannot be extended to the case with bounds on the links.

We now give another polynomial algorithm for the Bounded Network Restoration Problem with better worst case running time. We observed that the function $v(Y)$ is piecewise linear and convex in Y . Also, since the greedy procedure gives an integer solution to the linear relaxation of the formulation (4.3) whenever Y is integer, the optimal value of the integer program (4.3) and its linear relaxation coincide for integer Y . We can, therefore, devise a binary search algorithm for the BNR problem.

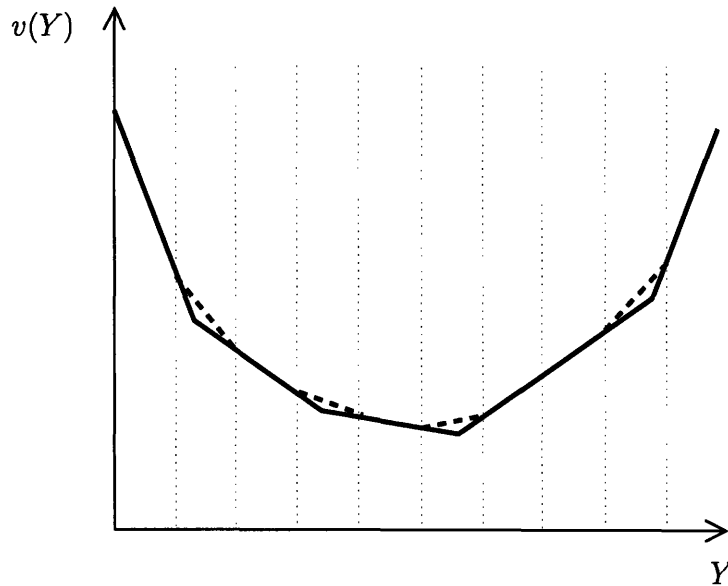


Figure 4-4: Slope used by the binary search algorithm

Figure 4-4 illustrates the function $v(Y)$. The grey vertical lines correspond to integer values of Y . Since we are interested only in integer solutions, we could use the slope shown with dotted lines for the binary search algorithm. The slopes are given by $v(Y) - v(Y - 1)$ for Y integer. For a binary search algorithm to work, we require that the set of Y values that we must consider to form a ‘small’ finite continuous interval. We show that this is the case.

Suppose there are k parallel links with nonnegative costs $c_1 \leq c_2 \leq \dots \leq c_k$, upper bounds n_1, n_2, \dots, n_k , and demands d_1, d_2, \dots, d_k . Let $d^* = \max_{1 \leq i \leq k} d_i$.

Lemma 4.2. *The values of Y for which the linear relaxation of the integer program (4.3) is feasible forms a continuous interval. Moreover, the value of Y that gives the optimal solution lies within $[\max(d^*, (\sum_{1 \leq i \leq k} d_i)/(k - 1)), \min(2d^*, \sum_{1 \leq i \leq k} n_i)]$. The linear relaxation is feasible for $Y = \min(2d^*, \sum_{1 \leq i \leq k} n_i)$.*

Proof. Given two feasible solutions (Y^1, y^1) and (Y^2, y^2) to the linear relaxation of the integer program (4.3), it is easy to see that $\frac{1}{2}(Y^1, y^1) + \frac{1}{2}(Y^2, y^2)$ is feasible for the relaxation. Therefore, the set of values of Y for which the linear relaxation is feasible is an interval.

Let i be the link for which $d_i = d^*$. Then the constraint $Y - y_i \geq d^*$ for the link i along with the nonnegativity condition $y_i \geq 0$ implies that Y must be at least d^* . Also, summing constraints (4.3a) for all links gives

$$kY - \sum_{1 \leq i \leq k} y_i \geq \sum_{1 \leq i \leq k} d_i.$$

Since $\sum_{1 \leq i \leq k} y_i = Y$, we conclude that $Y \geq (\sum_{1 \leq i \leq k} d_i)/(k - 1)$.

Summing the constraints (4.3b) for all links gives $Y \leq \sum_{1 \leq i \leq k} n_i$ and in any optimal solution, $y_i \leq d^*$ for every link i . Also, in an optimal solution, reducing y_i for any link i must render the solution infeasible. Therefore, for some link i , the constraint (4.3a) must be tight, implying that

$$Y = d_i + y_i \leq d^* + d^* = 2d^*.$$

Our assumption that the given BNR instance is feasible implies that the linear relaxation is feasible for $Y = \sum_{1 \leq i \leq k} n_i$. Since the optimal value of Y is at most $2d^*$, the relaxation is feasible for some value of Y not greater than $2d^*$. We know that the set of feasible Y values forms an interval, therefore, if $2d^* < \sum_{1 \leq i \leq k} n_i$, we conclude that the relaxation is feasible for $Y = 2d^*$. \square

From Lemma 4.2, we observe that the length of the interval that we need to search is bounded by a polynomial in the size of the input. We also observe that if for some value of Y in this interval, the linear relaxation is infeasible, the feasible interval lies to the right of Y . We already established a greedy procedure to identify the slope of the cost function at any integer value of Y . Therefore, we can perform binary search on the interval to obtain an optimal solution to the Bounded Network Restoration Problem. We present the complete binary search algorithm for the Bounded Network Restoration problem below:

BOUNDED NETWORK RESTORATION ALGORITHM()

Input: Parallel links $1, 2, \dots, k$, facility costs for links c_i , bounds n_i , and demands d_i .

Output: Number of facilities y_i on each link that minimizes total cost.

$$(1) \quad \underline{Y} := \max(d^*, \lceil (\sum_{1 \leq i \leq k} d_i) / (k - 1) \rceil), \quad \text{and} \quad \bar{Y} := \min(2d^*, \sum_{1 \leq i \leq k} n_i)$$

$$(2) \quad \Delta_l := -1, \text{ and } \Delta_r = -1$$

(4) **while** ($\Delta_l < 0$ or $\Delta_r < 0$)

$$(5) \quad Y := \lceil (\bar{Y} + \underline{Y}) / 2 \rceil$$

(6) Use greedy procedure to calculate $v(Y)$ and $y_i, i = 1, 2, \dots, k$.

(7) **if** Y is infeasible **then** $\Delta_l := 1, \Delta_r := -1$

$$(8) \quad \text{else } \Delta_l := v(Y - 1) - v(Y),$$

$$(9) \quad \Delta_r := v(Y + 1) - v(Y)$$

(11) **if** $\Delta_l < 0$ **then** $\bar{Y} := Y$

(12) **else** $\underline{Y} := Y$

(13)

(14) **return** $\{y_i, i = 1, 2, \dots, k\}$

Theorem 4.3. *The binary search algorithm solves the Bounded Network Restoration Problem correctly. The running time of the algorithm is $O(k \log d^*)$.*

Proof. In this algorithm, we start with an interval that contains the optimal value of Y . At each stage, we calculate right side slope Δ_r and left side slope Δ_l at the current value Y given by the dotted lines in Figure 4-4. We know that if the problem is infeasible with current value of Y , the feasible interval for Y must be to the right. The algorithm sets the slopes accordingly. We stop when the cost is increasing on both sides, which implies that the current solution is optimal.

At each stage of the algorithm, we use the greedy procedure to calculate the

optimal solution with the current value of Y and also evaluate optimal costs at $Y - 1$ and $Y + 1$. For a two node network with k parallel edges, this requires $O(k)$ time. Since we reduce the search interval for Y by half each time, the number of stages depends logarithmically on the size of the initial interval, which is bounded by $2d^*$, giving an overall running time of $O(k \log d^*)$. We note that checking both left and right side slopes at every stage (as we do in our algorithm) will improve the speed of the algorithm in practice, but does not change its worst case complexity. \square

4.3 Lower Bounds For The SCEP

We provide four different lower bounds for the SCEP. The first two lower bounds are used to develop a $(5 + \epsilon)$ -factor approximation algorithm for the SCEP. The other two lower bounds are straightforward, but useful as we develop another approximation algorithm for the SCEP with a better guarantee.

Least Cost Radial Links Lower Bound

When all the initial capacities are zero, that is, when the given network is empty, we can establish a simple lower bound that uses only the cost of the radial links. Let c_i^* be the smallest facility cost of all the radial links $R(i)$ out of switch i , and d_i^* be the maximum direct demand out of switch i .

Lemma 4.4. $\sum_{i \in \mathcal{T}} c_i^* \cdot \lceil d_i^*/C \rceil$ is a lower bound to the optimal cost of the SCEP with zero initial capacities.

Proof. Consider a feasible solution \mathbf{y} to the SCEP with cost Z . Let d_{ij} be an arbitrary demand. When the direct link $\langle i, j \rangle$ fails, failure protection implies that all of the demand d_{ij} will be carried through the hub. Therefore, the total number of facilities

on all the radial links out of switch i is at least $\lceil d_{ij}/C \rceil$ and so

$$\begin{aligned} \sum_{e \in R(i)} y_e &\geq \left\lceil \frac{d_{ij}}{C} \right\rceil, \quad \forall j \in \mathcal{T} \setminus \{i\}. \\ \Rightarrow \sum_{e \in R(i)} y_e &\geq \left\lceil \frac{d_i^*}{C} \right\rceil. \end{aligned}$$

Consequently,

$$\begin{aligned} Z &\geq \sum_{i \in \mathcal{T}} \sum_{e \in R(i)} c_e \cdot y_e \\ &\geq \sum_{i \in \mathcal{T}} c_e^* \cdot \left\lceil \frac{d_i^*}{C} \right\rceil. \end{aligned}$$

Since this is true for any feasible solution, it is true for any optimal solution. \square

Fail Safe Direct Links Lower Bound

We obtain the second lower bound by relaxing the failure protection constraint for the direct links, giving a version of the SCEP in which the direct links are fail safe. As we show in Section 4.4.1, when we use the Decentralized Routing Scheme on this relaxation, the obtained subproblems are more easily amenable to approximation.

Integral Demand Lower Bound

Given a demand d_{ij} and the facility capacity C , we can divide the demand into two parts. The integral demand

$$d_{ij}^I = C \cdot \left\lfloor \frac{d_{ij}}{C} \right\rfloor$$

and the residual demand

$$d_{ij}^R = d_{ij} - d_{ij}^I.$$

We define the integral demand SCEP as the given SCEP in which we replace the demands by their integral components. That is, for each origin-destination pair

$\langle i, j \rangle$, the demand in the new problem is d_{ij}^I . The network and all other parameters are the same. The optimal cost of this new problem is clearly a lower bound on the optimal cost of the SCEP. Since in this relaxation, all demands are integral multiples of the facility capacity C , we can scale all the demands by C and assume the facility capacity to be 1. The resulting relaxation is significantly easier to solve.

Residual Demand Lower Bound

Finally, similar to what we just did, we define the residual demand SCEP to be the problem in which the demands are the residual demands d_{ij}^R while all the other parameters are as in the given SCEP. The optimal cost of the residual demand SCEP is also a lower bound to the given SCEP.

Let y_e^I and y_e^R be the number of facilities installed on link i by a feasible solution to the integral demand SCEP and the residual demand SCEP respectively. We note that if the initial capacities of all the links in the network are zero, the solution

$$y_e = y_e^I + y_e^R \quad \forall e \in E$$

is feasible to the SCEP. In particular, if we choose an optimal solution for both the integral and residual SCEP, we obtain a feasible solution to the SCEP with cost $Z^I + Z^R$. Therefore, if we can solve the integral and residual SCEP, we obtain a 2-factor approximation algorithm for the SCEP. We use this idea to develop an approximation algorithm for the SCEP in Section 4.4.2.

4.4 Decentralized Routing Algorithms for Survivable Capacity Expansion

Assuming we start from an empty network (that is, the initial capacities on the edges are all zero), we provide two constant factor approximation algorithms for the SCEP. The main idea underlying both the algorithms is the Decentralized Routing (DR) scheme proposed in Section 3.1. The DR scheme decomposes SCEP into subproblems,

one for each TDM switch. A heuristic for the subproblem with a guarantee of α translates to a 2α -approximation algorithm for the SCEP. However, when the DR scheme is applied to the SCEP, the resulting subproblem is difficult to solve (even approximately). We use two different approaches to obtain subproblems that can be approximated well, each one yielding an approximation algorithm for the SCEP. The first algorithm has a performance guarantee of $(5 + \epsilon)$ and the second algorithm improves this worst case performance ratio to $(4 + \epsilon)$.

4.4.1 A $(5 + \epsilon)$ -approximation Algorithm

The outline of the algorithm we propose is as follows: We first solve the problem assuming that the direct links never fail. So we need to protect only against the failure of radial links. We call this problem SCEP with Failsafe Direct links (FD). We develop a $(4 + \epsilon)$ approximation algorithm for the FD problem using the Decentralized Routing Scheme. Finally, we convert a feasible solution to the FD problem into a feasible solution to the SCEP.

In our presentation of the algorithm, we first show how to convert a feasible solution to the FD problem to a feasible solution to the SCEP without increasing the cost very much. We then present our approximation algorithm for the FD problem.

Observation 4.5. *Let (\mathbf{x}, \mathbf{y}) be a feasible solution to the SCEP with failsafe direct links. Let the number of spare facilities on the radial links out of switch i , $\sum_{e \in R(i)} y_e - \lceil f_i/C \rceil$, be $s(i)$. Then, for each switch i , if we add $\left(\max_{j \in T \setminus \{i\}} x_{ij} - s(i)\right)$ facilities to the cheapest radial link out of i , the resulting solution is feasible for the SCEP.*

Proof. We need to show that when any direct link $\langle i, j \rangle$ fails, the network has enough additional capacity on the route $i \rightarrow \text{hub} \rightarrow j$. The available capacity (under the failure of the direct link $\langle i, j \rangle$) on the radial link out of switch i is

$$s(i) + \left(\max_{j \in T \setminus \{i\}} x_{ij} - s(i) \right) = \max_{j \in T \setminus \{i\}} x_{ij} \geq x_{ij}.$$

Therefore, when the link $\langle i, j \rangle$ fails, we can route all the demand on the direct

link through the hub. Since this is true for any direct link, the resulting solution is feasible for the SCEP. \square

We can use this result to convert any approximation algorithm for the FD to an approximation algorithm for the general SCEP. We show that the performance guarantee becomes slightly worse.

Proposition 4.6. *An α -approximation algorithm for the FD problem implies the existence of an $(\alpha + 1)$ -approximation for the SCEP for any network with zero initial capacities.*

Proof. Consider the following algorithm for the SCEP: We first assume that the direct links are failsafe, and obtain a feasible solution with cost Z^α by using the α -approximation algorithm for the SCEP with failsafe direct links. By Observation 4.5, we obtain a feasible solution to the SCEP with cost Z by adding the requisite number of facilities to the cheapest radial links out of each switch. Let c_i^1 be the cost per facility of the cheapest radial link out of switch i . Let Z^* and Z^{FD} be the optimal cost of the SCEP and the SCEP with failsafe direct links. Clearly, $Z^* \geq Z^{FD}$. Also, if d_i^* be the maximum direct demand out of switch i , we note that the number of facilities installed by a solution to the SCEP with failsafe direct links on a direct link out of switch i is at most $\lceil d_i^*/C \rceil$. Then,

$$\begin{aligned}
Z &= Z^\alpha + \sum_{i \in \mathcal{T}} c_i^1 \cdot \left(\max_{j \in \mathcal{T} \setminus \{i\}} x_{ij} - s(i) \right) \\
&\leq Z^\alpha + \sum_{i \in \mathcal{T}} c_i^1 \cdot \left(\lceil d_i^*/C \rceil - s(i) \right) \\
&\leq \alpha Z^{FD} + \sum_{i \in \mathcal{T}} c_i^1 \cdot \lceil d_i^*/C \rceil \\
&\leq \alpha Z^* + Z^* \\
&= (\alpha + 1) \cdot Z^*.
\end{aligned} \tag{4.4}$$

The inequality (4.4) is due to Lemma 4.4. Therefore the algorithm we outlined is an $(\alpha + 1)$ -factor approximation algorithm for the SCEP. \square

We now address the FD and develop an approximation algorithm for the problem. In the integer program (4.1), if we remove constraints (4.1e) that ensure that the network have enough capacity to route demands in case of the failure of direct links, we obtain the following formulation for the FD problem:

$$\begin{aligned}
\text{(FD)} \quad \text{Minimize} \quad & \sum_{i < j \in \mathcal{T}} c_{ij} \cdot x_{ij} + \sum_{i \in \mathcal{T}} \sum_{e \in R(i)} c_e \cdot y_e \\
& f_{ij}^d + f_{ij}^i = d_{ij}, & \forall i < j \in \mathcal{T} \quad (4.5a) \\
& f_{ij}^d \leq C \cdot x_{ij}, & \forall i < j \in \mathcal{T} \quad (4.5b) \\
& \sum_{j \in \mathcal{T} \setminus \{i\}} f_{ij}^h + d_i \leq f_i & \forall i \in \mathcal{T} \quad (4.5c) \\
& C \cdot \left(\sum_{f \in R(i)} y_f - y_e \right) \geq f_i, & \forall e \in R(i), \forall i \in \mathcal{T} \quad (4.5d) \\
& x_{ij} \in \mathbb{Z}^+, & \forall j \in \mathcal{T} \setminus \{i\} \\
& y_e \in \mathbb{Z}^+, & \forall e \in R(i) \\
& f_{ij}^d, f_{ij}^i \geq 0, & \forall i, j \in \mathcal{T} \\
& f_i \geq 0.
\end{aligned}$$

We use the Decentralized routing scheme to decompose the problem into a subproblem for each switch. In the subproblem for switch i , we would like to route the demands out of switch i either on the direct link, or send them to the *hub* node. The facility cost of the radial links are unchanged, but the cost of direct links are reduced by half. The subproblem for switch i is the following integer program:

$$\begin{aligned}
\text{(FD-}i\text{)} \quad \text{Minimize} \quad & \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij} + \sum_{e \in R(i)} c_e \cdot y_e \\
& f_{ij}^d + f_{ij}^i = d_{ij}, & \forall j \in \mathcal{T} \setminus \{i\} \quad (4.6a) \\
& f_{ij}^d \leq C \cdot x_{ij}, & \forall j \in \mathcal{T} \setminus \{i\} \quad (4.6b) \\
& \sum_{j \in \mathcal{T} \setminus \{i\}} f_{ij}^h + d_i \leq f_i & (4.6c)
\end{aligned}$$

$$\begin{aligned}
C \cdot \left(\sum_{f \in R(i)} y_f - y_e \right) &\geq f_i, & \forall e \in R(i) & \quad (4.6d) \\
x_{ij} &\in \mathbb{Z}^+, & \forall j \in \mathcal{T} \setminus \{i\} & \\
y_e &\in \mathbb{Z}^+, & \forall e \in R(i) & \\
f_{ij}^d, f_{ij}^i &\geq 0, & \forall i, j \in \mathcal{T} & \\
f_i &\geq 0. & &
\end{aligned}$$

We now make two observations that enable us to design a $(2 + \epsilon)$ -approximation algorithm for the subproblem FD- i .

Observation 4.7. *There is an optimal solution to the integer program FD- i in which the decision variable f_i is an integral multiple of the facility capacity C .*

Proof. Given an optimal solution to the integer program, we can replace f_i by $C \cdot \lceil f_i/C \rceil$ leaving all other variables unchanged. The new solution has the same cost. Since we increase the value of the variable f_i , the only constraint that could be violated is (4.6d). However, since the lefthand side of this constraint is an integral multiple of C , the constraint will be feasible for the new solution. The new solution satisfies the statement of the observation. \square

Let the cost of the Network Restoration problem on a parallel path network consisting of the radial links $R(i)$ with the demands of every link is d be $c^{NR}(d)$.

Observation 4.8. *In an optimal solution to the integer program FD- i , the total cost, $\sum_{e \in R(i)} c_e \cdot y_e$, of the facilities on the radial links $R(i)$ equals $c^{NR}(\lceil f_i/C \rceil)$.*

Proof. Given an optimal solution to the integer program FD- i , we first change the variable f_i to $\lceil f_i/C \rceil$. The proof of Observation 4.7 implies that the new solution is optimal too. Now, if we fix all the variables except the y_e variables to this solution, the integer program becomes a network restoration problem with demand $\lceil f_i/C \rceil$ on all the radial links. Since we started with an optimal solution to FD- i , the y_e values in this solution must be optimal to the Network Restoration problem. We conclude that the total cost of all the facilities on the radial links in an optimal solution to

the FD- i must be equal to the optimal cost of the Network Restoration problem, $c^{NR}(\lceil f_i/C \rceil)$. \square

Using Observations 4.7 and 4.8, we can write an equivalent formulation for FD- i as follows: In the integer program (4.6), we replace the variable f_i by $C \cdot y$, where y is an integer. We remove the constraints (4.6d), and modify the objective function so that the total cost of the facilities on the radial links $R(i)$ out of switch i is $c^{NR}(y)$. The new formulation is:

$$\text{Minimize} \quad \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij} + c^{NR}(y)$$

$$f_{ij}^d + f_{ij}^i = d_{ij}, \quad \forall j \in \mathcal{T} \setminus \{i\} \quad (4.7a)$$

$$f_{ij}^d \leq C \cdot x_{ij}, \quad \forall j \in \mathcal{T} \setminus \{i\} \quad (4.7b)$$

$$\sum_{j \in \mathcal{T} \setminus \{i\}} f_{ij}^h + d_i \leq C \cdot y \quad (4.7c)$$

$$x_{ij} \in \mathbb{Z}^+, \quad \forall j \in \mathcal{T} \setminus \{i\}$$

$$y \in \mathbb{Z}^+$$

$$f_{ij}^d, f_{ij}^i \geq 0, \quad \forall i, j \in \mathcal{T}.$$

The cost function in (4.7) is not linear since $c^{NR}(y)$ isn't a linear function of y . We now approximate the cost function $c^{NR}(y)$ by a linear function. We establish linear lower and upper bounds on $c^{NR}(y)$.

Lemma 4.9. *Let c^1 and c^2 be the least and the second least cost among links in a (unbounded) Network Restoration problem with the same demand d for all the links. Then $c^2 d$ is a lower bound on the optimal cost $c^{NR}(d)$ of this Network Restoration problem, and $(c^1 + c^2)d$ is an upper bound on the optimal cost.*

Proof. Let the links $\{1, 2, \dots, k\}$ be ordered in ascending order of facility cost $c^i, 1 \leq i \leq k$. Consider an arbitrary feasible solution y to the Network Restoration problem.

We must have

$$\sum_{2 \leq i \leq k} y_i \geq d,$$

implying that

$$\begin{aligned} c^{NR}(d) &= \sum_{1 \leq i \leq k} c^i y_i \\ &\geq \sum_{2 \leq i \leq k} c^i y_i \\ &\geq c^2 \cdot \sum_{2 \leq i \leq k} y_i \\ &\geq c^2 d. \end{aligned}$$

Therefore, $c^2 d$ is a lower bound to the optimal cost $c^{NR}(d)$.

We note that installing d facilities each on links 1 and 2 is feasible for the Network Restoration problem, implying that $(c^1 + c^2)d$ is an upper bound on $c^{NR}(d)$. \square

Figure 4-5 shows the optimal cost $c^{NR}(d)$ of the Network Restoration problem along with the lower and upper bounds we just established as a function of the demand d . An additional lower bound to the NR problem is the optimal cost $\underline{c}^{NR}(d)$ of its linear programming relaxation. When the y_i variables are allowed to take fractional values, it is possible to show (using results in Magnanti and Wang [38] and Wang [53]) that the optimal cost of the NR problem is a linear function of the demand d . Using the same argument provided in Lemma 4.9, we can see that $\underline{c}^{NR}(d) \geq c^2 d$. Figure 4-5 reflects this observation. We also note that the optimal cost $c^{NR}(d)$ is neither convex nor concave in the demand d . Since the variables y_i are integer, we need be interested only in integer values of d . So we show the values of $c^{NR}(d)$ only for integer d . The equally spaced vertical lines in the figure correspond to integer values of d .

We can therefore linearize the objective function in the integer program (4.6) to obtain the following approximate formulation:

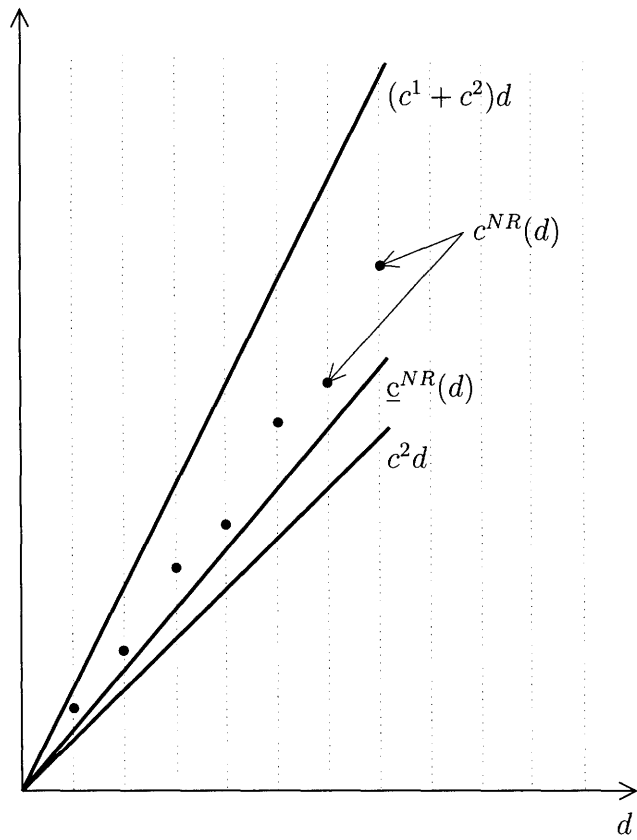


Figure 4-5: Optimal cost of the network restoration problems with upper and lower bounds

$$\begin{aligned}
\text{Minimize} \quad & \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij} + (c_i^1 + c_i^2) \cdot y \\
& f_{ij}^d + f_{ij}^i = d_{ij}, & \forall j \in \mathcal{T} \setminus \{i\} & \quad (4.8a) \\
& f_{ij}^d \leq C \cdot x_{ij}, & \forall j \in \mathcal{T} \setminus \{i\} & \quad (4.8b) \\
& \sum_{j \in \mathcal{T} \setminus \{i\}} f_{ij}^h + d_i \leq C \cdot y & & \quad (4.8c) \\
& x_{ij} \in \mathbb{Z}^+, & \forall j \in \mathcal{T} \setminus \{i\} & \\
& y \in \mathbb{Z}^+ & & \\
& f_{ij}^d, f_{ij}^i \geq 0, & \forall i, j \in \mathcal{T}. &
\end{aligned}$$

This is the Expandable Min-Knapsack Problem for which we developed a PTAS in Section 2.5.2 with a running time of $O(n^3/\epsilon)$. Given a feasible solution to Problem (4.8), we can obtain a feasible solution to Problem (4.7) by solving a Network Restoration Problem on the radial links out of switch i with demand y . We set the number of facilities on the radial links to be equal to the solution to the Network Restoration problem. As we argued in the proof of Observation 4.8, this solution is feasible for the FD- i . We present the complete algorithm below:

LINEAR COST APPROXIMATION ALGORITHM()

Input: Direct links $\langle i, j \rangle$, radial links $R(i)$, facility costs c_{ij} , facility capacity C , direct demands d_{ij} , and hub demand d_i .

Output: Number of facilities x_{ij} on direct links and y_e on radial links.

- (1) Formulate an Expandable Knapsack Problem with initial knapsack size 0 and a knapsack expansion cost $c_i^1 + c_i^2$
- (2) Find an $\frac{\epsilon}{2}$ -approximate solution (x, y) to the EKP
- (3) Solve a Network Restoration problem on the radial links with demand y on all links. Let \hat{y} be the optimal solution obtained
- (4) **return** $\{x, \hat{y}\}$

Lemma 4.10. *The Linear Cost Approximation algorithm is a $(2 + \epsilon)$ -factor approximation algorithm for the subproblem FD- i .*

Proof. Let (x^*, y^*) be an optimal solution to FD- i with cost Z^* . We know that for some \tilde{y} with (x^*, \tilde{y}) is feasible for (4.8),

$$Z^* = \sum_{j \in T \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij} + c^{NR}(\tilde{y}).$$

Let Z^{LC} be the optimal cost of the integer program (4.8). Then,

$$\begin{aligned} Z^* &= \sum_{j \in T \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij}^* + c^{NR}(\tilde{y}) \\ &\geq \sum_{j \in T \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij}^* + c_i^2 \cdot \tilde{y} && \text{(Lemma 4.8)} \\ &\geq \frac{1}{2} \left(\sum_{j \in T \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij}^* + (c_i^1 + c_i^2) \cdot \tilde{y} \right) && \text{(since } c_i^1 \leq c_i^2 \text{)} \\ &\geq \frac{1}{2} Z^{LC} && \left((x^*, \tilde{y}) \text{ is feasible to the IP (4.8)).} \right) \end{aligned}$$

Also, let Z be the cost of the solution (x, \hat{y}) returned by the Linear Cost Approximation Algorithm. And let (x, y) be the solution to IP (4.8) returned by the PTAS for the Expandable Knapsack Problem during the run of the LCA. We have

$$\begin{aligned} Z &= \sum_{j \in T \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij} + \sum_{e \in R(i)} c_e \cdot \hat{y}_e \\ &= \sum_{j \in T \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij} + c^{NR}(y) \\ &\leq \sum_{j \in T \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij} + (c_i^1 + c_i^2) \cdot y && \text{(Lemma 4.8)} \\ &\leq \left(1 + \frac{\epsilon}{2}\right) Z^{LC} && \left((x, y) \text{ is } \frac{\epsilon}{2}\text{-approximate} \right). \end{aligned}$$

Putting both together, we have

$$Z \leq \left(1 + \frac{\epsilon}{2}\right) Z^{LC} \leq \left(1 + \frac{\epsilon}{2}\right) \cdot 2 \cdot Z^* = (2 + \epsilon) Z^*.$$

□

Since we used the Decentralized Routing scheme on the problem FD to obtain the subproblem FD- i , a $(2 + \epsilon)$ -approximation algorithm for the FD- i implies a $(4 + \epsilon)$ -approximation algorithm for the FD (by applying Lemma 3.1). We can obtain a feasible solution (\hat{x}, \hat{y}) to the FD as follows: We set the number of facilities on all the radial links out of switch i to be equal to the number of facilities in the feasible solution to the subproblem FD- i produced by the Linear Cost Approximation algorithm. For the facilities on direct links, if x_{ij}^i and x_{ij}^j are the number of facilities installed on the direct link $\langle i, j \rangle$ by the Linear Cost Approximation algorithm on the FD- i and FD- j subproblems, we set $\hat{x}_{ij} = \max(x_{ij}^i, x_{ij}^j)$.

Finally, we can convert the feasible solution to the FD to a feasible solution to the SCEP by using Lemma 4.5. We present the complete algorithm for the SCEP, which we call Failsafe Direct Links Approximation Algorithm.

FAILSAFE DIRECT LINKS APPROXIMATION ALGORITHM()

Input: Direct links $\langle i, j \rangle$, radial links $R(i)$ for all switches i , facility costs for direct and radial links c_{ij} , facility capacity C , direct demands d_{ij} , and hub demand d_i .

Output: Number of facilities x_{ij} on direct links and y_e on radial links that minimizes total cost.

- (1) Ignore direct link failure conditions to formulate SCEP with Failsafe Direct Links (FD)
- (2) Decompose the problem obtained. Create subproblem FD- i for each switch i
- (3) Solve each subproblem FD- i using the Linear Cost Approximation Algorithm
- (4) Put solutions of subproblems together to obtain feasible solution (x, y) to FD
- (5) In this solution, let f_{ij}^h be the amount of demand d_{ij} routed through the hub
- (6) Let $s(i) := \sum_{e \in R(i)} C y_e - (d_i + \sum_{j \in T \setminus \{i\}} f_{ij}^h)$
- (7) **foreach** switch i
- (8) **foreach** e in $R(i)$
- (9) Let $\hat{y}_e := y_e + \max_{j \in T \setminus \{i\}} y_{ij} - s(i)$
- (10) **return** (x, \hat{y})

Theorem 4.11. *The Failsafe Direct Links Approximation Algorithm is an approxi-*

ation algorithm for the SCEP with a performance guarantee of $(5 + \epsilon)$.

Proof. This theorem follows directly from Lemma 4.10, Lemma 3.1, and Lemma 4.5. □

4.4.2 A $(4 + \epsilon)$ -approximation Algorithm

In this Section, we describe another approximation algorithm for the SCEP with zero initial capacities. This algorithm also uses the Decentralized Routing Scheme and has a better worst case performance guarantee of $4 + \epsilon$. We split each demand into two parts: an integral component which is a multiple of the facility capacity C , and a residual component that is less than the facility capacity C . We then solve two SCEP instances, one with all the integral demands and another with just the residual demands. We develop approximation algorithms based on the DR scheme for both cases, and show that putting these solutions together yields an approximation algorithm to the original SCEP.

Small Direct Demands

We assume that all the direct demands, i.e., demands between two TDM switches, are less than the facility capacity C . Therefore, we will never install more than one facility on a direct link. The total demand d_i between switch i and all softswitches can be arbitrary. We formulate this problem as the following integer program:

$$\begin{aligned}
 \text{(R)} \quad \text{Minimize} \quad & \sum_{i < j \in \mathcal{T}} c_{ij} \cdot x_{ij} + \sum_{i \in \mathcal{T}} \sum_{e \in R(i)} c_e \cdot y_e \\
 & \sum_{j \in \mathcal{T}} d_{ij} \cdot (1 - x_{ij}) + d_i \leq f_i, \quad \forall i \in \mathcal{T} \quad (4.9a)
 \end{aligned}$$

$$C \cdot \left(\sum_{f \in R(i)} (u_f + y_f) - (u_e + y_e) \right) \geq f_i, \quad \forall e \in R(i), \forall i \in \mathcal{T} \quad (4.9b)$$

$$\sum_{e \in R(i)} (u_e + y_e) \geq 1, \quad \forall i \in \mathcal{T} \quad (4.9c)$$

$$\begin{aligned}
f_i &\geq 0, & \forall i \in \mathcal{T} \\
x_{ij} &\in \{0, 1\}, & \forall i < j \in \mathcal{T} \\
y_e &\in \mathbb{Z}^+, & \forall e \in R(i), \forall i \in \mathcal{T}.
\end{aligned}$$

We highlight the difference between the formulation (4.1) for the general SCEP and the formulation (4.9) when the demands are small. First, the variable x_{ij} is binary since we will never install more than one facility on a direct link between two TDM switches. Second, we do not explicitly consider situations when a direct link fails. Since the demands are less than C , the spare capacity required on the radial links to handle traffic rerouted from a failed direct link is no more than C . It is easy to see that the constraint (4.9b) ensures that the spare capacity on radial links out of switch i is at least C if $f_i > 0$. If $f_i = 0$, all demands are routed on the direct links, and constraint (4.9c) ensures that there is at least one facility on the radial links incident to the TDM switch i .

We apply the Decentralized Routing scheme for this problem as follows: we duplicate each demand d_{ij} between two TDM switches i and j into directed demands D_{ij} and D_{ji} each with the same magnitude as d_{ij} . For each TDM switch i , we consider the problem of routing all demands out of switch i either on the direct link to the destination, or to the hub. This subproblem for switch i is the following:

$$\begin{aligned}
\text{(R-}i\text{)} \quad \text{Minimize} \quad & \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij} + \sum_{e \in R(i)} c_e \cdot y_e \\
& \sum_{j \in \mathcal{T} \setminus \{i\}} d_{ij} \cdot (1 - x_{ij}) + d_i \leq f_i & (4.10a)
\end{aligned}$$

$$C \cdot \left(\sum_{f \in R(i)} (u_f + y_f) - (u_e + y_e) \right) \geq f_i, \quad \forall e \in R(i) \quad (4.10b)$$

$$\sum_{e \in R(i)} (u_e + y_e) \geq 1 \quad (4.10c)$$

$$\begin{aligned}
x_{ij} &\in \{0, 1\}, & \forall j \in \mathcal{T} \setminus \{i\} \\
y_e &\in \mathbb{Z}^+, & \forall e \in R(i) \\
f_i &\geq 0.
\end{aligned}$$

For every switch i , let $\{\hat{x}_{ij}, \hat{y}_e \mid j \in \mathcal{T} \setminus \{i\}, e \in R(i)\}$ be a feasible solution to $SPa - i$. We consider the solution (x, \hat{y}) with $x_{ij} = \max(\hat{x}_{ij}, \hat{x}_{ji})$ for every pair of TDM switches i and j . Lemma 3.1 shows that the solution (x, \hat{y}) is feasible for the integer program R and has cost at most twice that of (\hat{x}, \hat{y}) .

We can use an α -approximation algorithm for the subproblem $R-i$ to obtain a 2α -approximation algorithm for the SCEP. We now provide a Polynomial Time Approximation Scheme for the subproblem $R-i$. This directly implies a $(2+\epsilon)$ -approximation algorithm for the SCEP with small demands.

We observe that in the subproblem $R-i$, the left hand side of constraint (4.10b) is an integral multiple of C . Therefore, if we define a new integer variable z_i and replace f_i by Cz_i in constraints (4.10b) and (4.10c), the optimal cost of the subproblem does not change. We solve this modified subproblem instead of $R-i$.

Since the demands are all less than C , the set of values of the variable z_i we need to consider are $\lceil d_i/C \rceil, \lceil d_i/C \rceil + 1, \dots, \lceil d_i/C \rceil + n$. We provide a solution procedure for the subproblem when the value of z_i is known, and solve the subproblem for each of the $n + 1$ possible values of z_i .

We observe that when z_i is fixed, the subproblem decomposes into two problems. After rearranging the variables and parameters in the constraints, we write the first problem as the following min-knapsack problem:

$$\begin{aligned}
(X) \quad \text{Minimize} \quad & \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij} \\
& \sum_{j \in \mathcal{T} \setminus \{i\}} d_{ij} \cdot x_{ij} \geq \left(\sum_{j \in \mathcal{T} \setminus \{i\}} d_{ij} \right) + d_i - Cz_i & (4.11a) \\
& x_{ij} \in \{0, 1\}, & \forall j \in \mathcal{T} \setminus \{i\}.
\end{aligned}$$

While the min-knapsack problem is known to be NP-Hard, we can use a polynomial

time approximation scheme for the problem (see Gens and Levner [24]) to obtain an ϵ -approximate solution for any positive ϵ . We noted in Section 2.5.1 that this can be done in $O(n^2/\epsilon)$ time.

The second subproblem is the following mathematical program:

$$(Y) \quad \text{Minimize} \quad \sum_{e \in R(i)} c_e \cdot y_e$$

$$\left(\sum_{f \in R(i)} y_f \right) - y_e \geq z_i - \left(\sum_{f \in R(i)} u_f \right) - u_e, \quad \forall e \in R(i) \quad (4.12a)$$

$$\sum_{e \in R(i)} (u_e + y_e) \geq 1 \quad (4.12b)$$

$$y_e \in \mathbb{Z}^+, \quad \forall e \in R(i).$$

Constraint (4.12a) is a rearranged version of constraint (4.10b). We can ignore constraint (4.12b) when we solve this problem, and if the resulting optimal solution y violates this constraint (this will happen only if all the u_e 's and all y_e 's are zero), we set $y_e = 1$ for the edge e with the cheapest cost c_e . We let $d_e = \max(0, z_i - (\sum_{f \in R(i)} u_f) + u_e)$. Solving the problem (Y) is equivalent to solving a Network Restoration Problem with demands d_e on the edges. Using the algorithm presented by Magnanti and Wang [38], we can solve this problem in $O(n)$ time.

Putting everything together, the complexity of our procedure for obtaining an ϵ -approximate optimal solution to the subproblem R- i is $O(n^3/\epsilon)$. By using this procedure in the Decentralized Routing scheme, we obtain a $(2 + \epsilon)$ -approximation algorithm for the SCEP when the direct demands are all less than the facility capacity C .

Integral Direct Demands

We now study the Survivable Capacity Expansion Problem in which every demand d_{ij} between two TDM switches i and j is an integral multiple of the facility capacity C . We do not restrict the demands between a TDM switch and a softswitch to be integral multiples of C . Without loss of generality, we assume that the initial capacity

u_{ij} on the links between two TDM switches i and j satisfy the constraint $d_{ij} \geq C u_{ij}$. We show that in this case, the demands can be scaled down in such a way that the facility capacity can be assumed to be 1. We first make the following observation:

Observation 4.12. *The SCEP with integral direct demands has an optimal solution in which the amount of demand d_{ij} between every pair of TDM switches i and j routed on the direct link $\langle i, j \rangle$ is an integral multiple of C .*

Proof. In any feasible solution, the total capacity (initial + added) is an integral multiple of C . If in the given optimal solution, for any demand $\langle i, j \rangle$, the amount of demand d_{ij} routed directly is not a multiple of C , we can increase this amount to next multiple of C without increasing the overall cost. \square

Consider a solution to the SCEP that satisfies the property stated in Observation 4.12. Since all the direct demands are integral multiples of C , in this solution the amount of direct demand d_{ij} routed through the hub is also an integral multiple of C . Therefore, for every TDM switch i , we can round the total demand d_i between the switch i and all the softswitches up to $\lceil d_i/C \rceil C$ without increasing the optimal cost.

Proposition 4.13. *In the SCEP with integral direct demands, for every TDM switch i , the total demand d_i between switch i and all the softswitches can be increased to the next integral multiple of C without increasing the optimal cost.*

In this problem, all the demands are integral multiples of C , so we can scale them down by the factor C and reduce the facility capacity to 1. Let \hat{d}_{ij} and \hat{d}_i be the

scaled demands. We can formulate the SCEP as the following integer program:

$$\begin{aligned}
\text{(I) Minimize } & \sum_{i < j \in \mathcal{T}} c_{ij} \cdot x_{ij} + \sum_{i \in \mathcal{T}} \sum_{e \in R(i)} c_e \cdot y_e \\
& u_{ij} + x_{ij} + f_{ij}^h = \hat{d}_{ij}, & \forall i < j \in \mathcal{T} & \quad (4.13a) \\
& \sum_{j \in \mathcal{T} \setminus \{i\}} f_{ij}^h + \hat{d}_i \leq z_i, & \forall i \in \mathcal{T} & \quad (4.13b) \\
& \sum_{f \in R(i)} (u_f + y_f) - (u_e + y_e) \geq z_i, & \forall e \in R(i), \forall i \in \mathcal{T} & \quad (4.13c) \\
& \sum_{f \in R(i)} (u_f + y_f) \geq z_i + u_{ij} + x_{ij}, & \forall i, j \in \mathcal{T} & \quad (4.13d) \\
& y_e \in \mathbb{Z}^+, & \forall e \in R(i), \forall i \in \mathcal{T} \\
& x_{ij} \in \mathbb{Z}^+, f_{ij}^h \geq 0, & \forall i < j \in \mathcal{T} \\
& z_i \in \mathbb{Z}^+, & \forall i \in \mathcal{T}.
\end{aligned}$$

We can use the Decentralized Routing scheme on formulation (4.13) to obtain the following subproblem for TDM switch i :

$$\begin{aligned}
\text{(I-}i\text{) Minimize } & \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij} + \sum_{e \in R(i)} c_e \cdot y_e \\
& u_{ij} + x_{ij} + f_{ij}^h = \hat{d}_{ij}, & \forall j \in \mathcal{T} \setminus \{i\} & \quad (4.14a) \\
& \sum_{j \in \mathcal{T} \setminus \{i\}} f_{ij}^h + \hat{d}_i \leq z_i, & & \quad (4.14b) \\
& \sum_{f \in R(i)} (u_f + y_f) - (u_e + y_e) \geq z_i, & \forall e \in R(i) & \quad (4.14c) \\
& \sum_{f \in R(i)} (u_f + y_f) \geq z_i + u_{ij} + x_{ij}, & \forall j \in \mathcal{T} \setminus \{i\} & \quad (4.14d) \\
& y_e \in \mathbb{Z}^+, & \forall e \in R(i) \\
& x_{ij} \in \mathbb{Z}^+, f_{ij}^h \geq 0, & \forall j \in \mathcal{T} \setminus \{i\} \\
& z_i \in \mathbb{Z}^+.
\end{aligned}$$

We now formulate the subproblem (4.14) as a Bounded Network Restoration prob-

lem. It is possible to eliminate the variables f_{ij}^h and z_i in the integer program (4.13) to obtain a formulation that can be interpreted as a BNR problem. However, we take a more intuitive approach. We observe that the objective of the subproblem is to route all the demands on one or more of the links incident to switch i . For the purpose of routing demands, we do not differentiate between a radial link or a direct link even though radial links can carry any demand, while direct links can only carry the demand between its endpoints. We address this difference by specifying an upper bound, equal to the demand d_{ij} , on the number of facilities that can be installed on any direct link $\langle i, j \rangle$. Let $D_i = \hat{d}_i + \sum_{j \in \mathcal{T} \setminus \{i\}} \hat{d}_{ij}$. Also let U_e be the sum of the initial capacities on all the links out of switch i except the link e . That is, $U_e = \sum_{f \in R(i)} u_f + \sum_{j \in \mathcal{T} \setminus \{i\}} u_{ij} - u_e$. The following is a BNR formulation for the SCEP:

$$\begin{aligned}
\text{(BNR-}i\text{)} \quad \text{Minimize} \quad & \sum_{j \in \mathcal{T} \setminus \{i\}} \frac{c_{ij}}{2} \cdot x_{ij} + \sum_{e \in R(i)} c_e \cdot y_e \\
& \sum_{f \in R(i)} y_f + \sum_{j \in \mathcal{T} \setminus \{i\}} x_{ij} - y_e \geq D_i - U_e, \quad \forall e \in R(i) \quad (4.15a)
\end{aligned}$$

$$\sum_{f \in R(i)} y_f + \sum_{k \in \mathcal{T} \setminus \{i\}} x_{ik} - x_{ij} \geq D_i - U_{ij}, \quad \forall j \in \mathcal{T} \setminus \{i\} \quad (4.15b)$$

$$\begin{aligned}
0 \leq x_{ij} \leq d_{ij} - u_{ij}, x_{ij} \in \mathbb{Z}^+ & \quad \forall j \in \mathcal{T} \setminus \{i\} \\
y_e \geq 0, y_e \in \mathbb{Z}^+ & \quad \forall e \in R(i).
\end{aligned}$$

Proposition 4.14. *The formulation (4.15) solves the subproblem (4.14).*

Proof. It is easy to verify that the constraints (4.15a) and (4.15b) are obtained from constraints (4.14a), (4.14b), (4.14c), and (4.14d) by eliminating the variables f_{ij}^h and z_i . The constraint $f_{ij}^h \geq 0$ implies that $x_{ij} \leq d_{ij} - u_{ij}$. We include this in the new formulation. \square

We can solve the problem (4.15) using the binary search algorithm presented in Section 4.2. Let $\{\hat{x}_{ij}, \hat{y}_e \mid j \in \mathcal{T} \setminus \{i\}, e \in R(i)\}$ be a solution to the integer program I- i produced by the binary search algorithm. The Decentralized Routing Algorithm

returns the solution (x, \hat{y}) with $x_{ij} = \max(\hat{x}_{ij}, \hat{x}_{ji})$ for every pair of TDM switches i and j . We show that this solution is feasible to the SCEP with at most twice the optimal cost.

Theorem 4.15. *The Decentralized Routing Algorithm is a 2-approximation algorithm for the SCEP with integral direct demands.*

Proof. This follows directly from Lemma 3.1 since we have developed a polynomial time algorithm for the subproblem I- i . \square

SCEP With Zero Initial Capacities

For every $\langle i, j \rangle$, let $d_{ij} = q_{ij}C + r_{ij}$ with $q_{ij} \geq 0$ and $r_{ij} < C$. Let $P(d), P(q)$, and $P(r)$ refer to the Survivable Capacity Expansion Problems on the graph G with demands d_{ij} , $q_{ij}C$, and r_{ij} , and let Z_d, Z_q , and Z_r denote the optimal cost of these problems. Clearly, $Z_d \geq Z_q$ and $Z_d \geq Z_r$. Let (x^q, y^q) and (x^r, y^r) be feasible solutions to $P(q)$, and $P(r)$. Let (x^d, y^d) be such that $x_{ij}^d = x_{ij}^q + x_{ij}^r$ for every pair of TDM switches i and j , and $y_e^d = y_e^q + y_e^r$ for every radial link e . It is easy to see that (x^d, y^d) is a feasible solution to $P(d)$.

This suggests the following heuristic for the SCEP on a graph G with zero initial capacity on all links. Split each demand into an integral multiple of C and a residual. Using the approximation algorithms presented in Sections 4.4.2 and 4.4.2, solve two subproblems, one with all demands as integral multiple of C , and another in which every demand is smaller than C . Let (x^q, y^q) and (x^r, y^r) be the solutions to the two subproblems returned by the approximation algorithms. The solution to the SCEP returned by the algorithm is $(x^d = x^q + x^r, y^d = y^q + y^r)$.

The cost of the solution (x^d, y^d) is

$$\begin{aligned} \text{cost}(x^d, y^d) &= \text{cost}(x^q, y^q) + \text{cost}(x^r, y^r) \\ &\leq 2 \cdot Z^q + \left(2 + \frac{\epsilon}{2}\right) \cdot Z^r \\ &\leq \left(2 + \frac{\epsilon}{2}\right) \cdot (Z^d + Z^d) \\ &= (4 + \epsilon) \cdot Z^d \end{aligned}$$

implying that the procedure we just outlined is a $(4+\epsilon)$ -approximation for the SCEP on a graph with no initial capacity.

Note that the algorithms we designed for both special cases (small demands and integral demands) are applicable even for problems with initial capacities on the links. However, in the presence of initial capacities, we cannot combine the solution of these two problems to obtain a feasible solution to the SCEP. Therefore, our approximation algorithm would not be applicable to SCEP with initial capacities on the links.

Chapter 5

Network Planning With Capacity Sharing

Current telecommunication networks measure demand data independently for each origin-destination pair. Consequently, when providers plan the network, they might overestimate the required capacity. This possibility becomes especially prevalent if the network is equipped with advanced routing technologies like Dynamic Routing that decides call routes as the call arrives based on the conditions of the network. We consider the Capacity Sharing Problem (CSP) that allows a planner to use temporal demand information (for example, which demand peaks occur simultaneously) in designing the network. We develop a general model to capture demand information, and present theoretical results for the CSP. We also propose a cutting plane based heuristic for the CSP.

5.1 The Capacity Sharing Problem

Consider a network $G = (V, E)$, with a set V of nodes and a set E of edges. Consistent with telecommunication applications, we assume that the network has undirected edges. Capacity installed on an edge could be used to send flow in both directions as long as the total flow on the edge in both directions is within the installed capacity on the edge. We also assume, for the sake of simplicity, that capacity can be installed

only in multiples of a single facility type of size C , and the cost per facility on edge e is c_e . The set of all origin destination pairs (or commodities) is K , and d_k is the demand of commodity k . Under these assumptions, we can formulate the Capacity Expansion Problem (CEP) as the following mathematical program:

$$\begin{aligned}
\text{(CEP)} \quad & \text{Minimize} \quad \sum_{e \in E} c_e \cdot y_e \\
& \sum_{(i,j) \in A} f_{ij}^k - \sum_{(j,i) \in A} f_{ji}^k = \begin{cases} d_k & \text{if } i = o(k) \\ -d_k & \text{if } i = \delta(k) \\ 0 & \text{otherwise.} \end{cases} \quad \forall i \in V, \forall k \in K \quad (5.1a) \\
& \sum_{k \in K} (f_{ij}^k + f_{ji}^k) \leq u_e + C \cdot y_e, \quad \forall e = \langle i, j \rangle \in E \quad (5.1b) \\
& y_e \in \mathbb{Z}_+, \quad \forall e \in E \\
& f_{ij}^k, f_{ji}^k \geq 0, \quad \forall e = \langle i, j \rangle \in E, \forall k \in K.
\end{aligned}$$

In this integer program, A is the set of all arcs. That is, A contains two directed arcs for every edge in E . K is the set of all origin destination pairs with demand, and the demand for O-D pair k with origin $o(k)$ and destination $\delta(k)$ is d_k . The parameter u_e is the current available capacity on edge e . The decision variable y_e is the number of facilities to be installed on the edge e .

The Capacity Sharing Problem (CSP) seeks to plan the network for a set of demands U that occur at different points in time. If we are to plan based on a single demand vector, we must use the peak demand for each origin-destination pair. Planning for more than one demand allows for more economic allocation of capacity since we might be able to exploit negative correlations between demands. In this problem, we are not concerned with the actual routing of the demand. We assume a network processor will use capacity in the network to dynamically find a feasible routing. This assumption is consistent with the ‘Dynamic Routing’ technology now prevalent among telecommunication companies.

Therefore, in the Capacity Sharing Problem, the objective is to identify a least cost capacity expansion plan so that all demands in a given set U have a feasible

routing. Since the set U is intended to represent the demand (of all origin destination pairs) at all points in time, we refer to the set U as the demand variation set. We consider several models for the demand variation set U , and propose a general realistic model. To solve this problem, we present a necessary and sufficient condition for a set of edge capacities to have a feasible routing for every demand in a given variation set U . We then use this condition to develop a computational procedure using a cutting plane approach for heuristically solving the Capacity Sharing Problem.

The so called Network Loading Problem is a special case of the Capacity Expansion Problem when the initial capacity u_e of all the edges in the network is zero. If there is a fixed cost for adding capacity, the CEP becomes the Capacitated Network Design problem. We collectively refer to all three problems as Network Planning problems. The results we establish for the capacity expansion problem are valid for all Network Planning problems and our cutting plane heuristic can be adapted for a capacity sharing version of the network design problem. Whether the procedure works well for network design problems must be ascertained through computational experiments that we will not undertake in this thesis.

The capacity sharing problem is closely related to the Robust Capacity Expansion problem under demand uncertainty. In the latter problem, we attempt to solve a Capacity Expansion Problem with a given, but uncertain, nominal demand. The set of possible values this demand can assume is given as an uncertainty set. The objective is to find a minimum cost capacity expansion plan that admits a feasible routing for all the demands in the uncertainty set. Mathematically, both the Capacity Sharing problem and the Robust Capacity Expansion problem are the same. But, as we discuss in Section 5.2, the demand variations (or uncertainty sets) that make practical sense differ for the two problems.

Finally, we also make note of the similarity between the Capacity Sharing problem and the Network Restoration Problem (NRP). In the NRP, the objective is to add spare capacity in the network to protect against single link failures. In a variant of the NRP called line restoration, when a link fails, the flow on the link is rerouted using spare capacity on a path from the tail to the head of the failed link. That is, the

failure of a link creates a ‘demand’ whose origin and destination are the tail and head of the link respectively. In this instance, since we are protecting against single link failure, we are assuming that no two demands are positive at the same time. In this respect, the NRP is a special case of the Capacity Sharing problem. However, when viewed as a Capacity Sharing Problem, the NRP has a crucial additional requirement: the demand due to the failure of a link $\langle i, j \rangle$ cannot be routed on the link $\langle i, j \rangle$. In spite of this difference, we believe that some of the ideas that are useful for addressing one of these problems, would be useful for the other. In fact, the heuristic we propose for the Capacity Sharing Problem is similar to the heuristic of Sakauchi, Nishimura, and Hasegawa [46] for the NRP.

5.2 Modeling Time Variations of Demand

In a telecommunication network the call demand between origin destination pairs varies with time. Let $d(t)$ be the vector of all origin-destination pair demands at observation time t , and let T be the set of all observation times. If we want to plan the network so that no demand is lost, we would then have to ensure that the network has enough capacity so that there is a feasible call routing for the demand $d(t)$ for all observation times t in T . The input for the network planning problem must include the set of all demands for which we would like to plan. One possibility is to use a list of viable demands, which we call ‘discrete demand variation’. In situations when it is not practical to use a list of all viable demand vectors, we seek a compact representation that is also flexible enough to allow us to capture all the demands with reasonable accuracy.

We list a few possible models for demand variation.

- 1. Discrete Demand Variation:** We are given a set with many different demand vectors. Each vector in the set could correspond to the peak demand for some (possibly more than one) origin destination pair.

2. Cardinality Restricted Demand Variation: Every commodity (origin - destination pair) k has a mean demand $\bar{d}_k \geq 0$ and a possible excess $h_k \geq 0$ beyond the mean. At any time at most Γ commodities could have a demand in excess of their mean. That is, the set of all viable demands is given by

$$U_C = \left\{ d \in \mathbb{R}^K : \sum_{k \in \bar{K}} \lceil |d_k - \bar{d}_k| / h_k \rceil \leq \Gamma, \bar{d} \leq d \leq \bar{d} + h \right\}.$$

Here, \bar{K} is the set of commodities k with $h_k > 0$.

3. Budget Restricted Demand Variation: As in the cardinality restricted case, every commodity has a mean as well as a possible excess. However, the excess demands must satisfy a constraint limiting a weighted sum of the excesses. Let v_k , $k \in K$ and w be nonnegative integers, then the budget variation set U_B is given by

$$U_B = \left\{ d \in \mathbb{R}^K : \sum_{k \in K} v_k (d_k - \bar{d}_k) \leq w, \bar{d} \leq d \leq \bar{d} + h \right\}.$$

Variations 2 and 3 are popular models for demand uncertainty in the robust optimization literature (see [14] and [7]). While modeling uncertainty, these models usually allow the uncertain parameter (in this case, the demand) to vary both above and below the mean. However, for a network planning problem, we can ignore the variation of demand below the mean. These demands are dominated by the mean demand and are therefore irrelevant for planning the network.

While the Cardinality Restricted Demand Variation is a good model for random uncorrelated demand, it is not realistic for modeling the variation of demand in a telecommunication network. This model treats all commodities as being independent of each other. But when we model time variation of demands, we expect a correlation between the demands of two commodities whose origins as well as destinations are in the same geographical area. The Budget Restricted Demand Variation suffers from the same drawback, but we can slightly improve the model by imposing a budget

constraint for each source node in the network.

$$U_B = \left\{ d \in \mathbb{R}^K : \sum_{k \in K: o(k)=i} v_k(d_k - \bar{d}_k) \leq w_i, \forall i \in V, \bar{d} \leq d \leq \bar{d} + h \right\}.$$

The following model explicitly captures the correlation between demands.

4. Subset Demand Variation: Each commodity k has a mean demand \bar{d}_k and a peak demand $\bar{d}_k + h_k$. We are given a collection of subsets K_1, K_2, \dots, K_t of the commodity set K . All commodities in a single subset peak at the same time, and all other commodities remain at or below their mean at this time. Therefore, the subset demand variation set U_S consists of t demands, and the demand d^i for the subset K_i is given by

$$d_k^i = \begin{cases} \bar{d}_k + h_k & \text{if } k \in K_i \\ \bar{d}_k & \text{otherwise.} \end{cases}$$

We now seek a general, more realistic, and practical model that includes these three as special cases. We start with the following two straightforward observations.

Observation 5.1. *If a solution $\{y_e : e \in E\}$ to a network planning problem is feasible for a demand vector d^1 , then it is feasible for all demands in the K -dimensional box $\{0 \leq d_k \leq d_k^1, \forall k \in K\}$.*

Proof. The observation follows from the fact that if $\{y_e : e \in E\}$ is feasible for d^1 , then it is feasible for all demands $d \leq d^1$. \square

Observation 5.2. *If a solution $\{y_e : e \in E\}$ to a network planning problem is feasible for two demand vectors d^1 and d^2 , then it is feasible for all demands in the convex combination of d^1 and d^2 .*

Proof. Let $\{(f^1)_{ij}^k : (i, j) \in A\}$ and $\{(f^2)_{ij}^k : (i, j) \in A\}$ be the flow variables that are feasible for the demands d^1 and d^2 respectively and for any $0 < \lambda < 1$ consider the demand vector $d^\lambda = \lambda d^1 + (1 - \lambda)d^2$. Define f^λ as follows:

$$(f^\lambda)_{ij}^k = \lambda(f^1)_{ij}^k + (1 - \lambda)(f^2)_{ij}^k, \forall (i, j) \in A, \forall k \in K.$$

It is easy to verify that f^λ satisfies flow conservation with the demand d^λ . Also, for any edge $e = \langle i, j \rangle$,

$$\begin{aligned}
\sum_{k \in K} (f^\lambda)_{ij}^k + (f^\lambda)_{ji}^k &= \sum_{k \in K} \left(\lambda (f^1)_{ij}^k + (1 - \lambda) (f^2)_{ij}^k + \lambda (f^1)_{ji}^k + (1 - \lambda) (f^2)_{ji}^k \right) \\
&= \lambda \cdot \sum_{k \in K} \left((f^1)_{ij}^k + (f^1)_{ji}^k \right) + (1 - \lambda) \sum_{k \in K} \left((f^2)_{ij}^k + (f^2)_{ji}^k \right) \\
&\leq \lambda y_e + (1 - \lambda) y_e \\
&= y_e.
\end{aligned} \tag{5.2}$$

The inequality (5.2) follows from the fact that the flows f^1 and f^2 are feasible for the solution $\{y_e : e \in E\}$. Since this is true for any λ in the interval $(0, 1)$, the observation follows. \square

Given a discrete set U , let $\mathbf{conv}(U)$ be its convex hull. We have the following consequence of Observation 5.2.

Corollary 5.3. *The network planning problem with capacity sharing with a discrete demand variation set U is equivalent to the problem with the demand set $\mathbf{conv}(U)$.*

Corollary 5.3 implies that we can replace any given demand variation set by the convex hull of all the points in the set. This leads us to the following more general model of demand variation, that includes as subcases all the three models outlined above.

5. Polyhedral Demand Variation: The set of all demands to be considered is given by a polyhedron. We make two assumptions about the polyhedron, both without loss of generality in our problem context.

1. Whenever a demand vector d' is in the variation set, all the demands $d \leq d'$ are also in the set. In particular, the zero demand vector, $\{d_k = 0, \forall k \in K\}$, is always in the variation set. Observation 5.1 allows us to make this modification to the demand variation set if required without changing the problem.

2. The given polyhedron is bounded. If not, the network planning problem will be infeasible.

For some positive integer t , a $|K| \times t$ matrix V , and a k -dimensional vector w , the demand variation polyhedron U_P is represented by

$$U_P = \left\{ d \in \mathbb{R}_+^{|K|} : Vd \leq w \right\}.$$

We can assume that the coefficient matrix V is nonnegative. This assumption restricts the set of all polyhedra that we consider for the following reason: we are confined to the space $\mathbb{R}_+^{|K|}$ of the commodities and we allow only less than or equal to constraints. It is easy to verify that we can represent all polyhedra that satisfy the two assumptions in this form.

In addition to being a very general model to capture time variation of demand, the polyhedral model is also practical. A telecommunication company could use the demand vectors from a limited number of measurements (possibly only during expected busy hours on some of the busiest days of the year). We could then use the convex hull of these demand vectors as our variation set. We note that when the number of such measurements is small, we could use all these demand vectors directly as a discrete demand variation. However, a polyhedral model allows a planner to impose additional constraints on the viable demands. For example, the planner could limit the total demand out of a switch, or the total demand within a region.

Hereafter, we assume that the demand variability is given as a polyhedron. The results we establish, and the heuristic we propose are also applicable when the demand variability is given as a discrete set.

5.3 Capacity Sharing: Modeling and Complexity

5.3.1 A Condition For Feasibility

We present a necessary and sufficient condition for a vector of edge capacities to have a feasible flow for every demand in a given variation set. This characterization is useful

because the condition could be used as constraints in a mathematical programming model for the Capacity Sharing Problem.

We use the following feasibility condition for the existence of a multicommodity flow (see Schrijver [49]). In the statement of this result, ℓ is an integer (length) function defined on the edges of the network and $dist_\ell(i, j)$ is the shortest distance between nodes i and j with respect to the length function ℓ .

Theorem 5.4. (Onaga [43]) *A set of edge capacities $\{u_e : e \in E\}$ has a feasible multicommodity flow for a demand vector $\{d_k : k \in K\}$ if and only if*

$$\sum_{e \in E} \ell_e u_e \geq \sum_{k \in K} d_k \cdot dist_\ell(o(k), \delta(k)) \quad (5.3)$$

for each length function $\ell : E \rightarrow \mathbb{Z}_+$.

In the context of the Capacity Sharing Problem, we have the following variant of this result.

Theorem 5.5. *A set of edge capacities $\{u_e : e \in E\}$ has a feasible call routing for every demand vector in the variation set $Vd \leq w$ if and only if*

$$\sum_{e \in E} \ell_e u_e \geq \max_{d: Vd \leq w} \left(\sum_{k \in K} d_k \cdot dist_\ell(o(k), \delta(k)) \right) \quad (5.4)$$

for each length function $\ell : E \rightarrow \mathbb{Z}_+$.

Proof. Clearly, u is feasible for all demands d in the polyhedron $Vd \leq w$ if and only if the relationship (5.3) is satisfied for the maximum righthand side. \square

Let u_e be the initial capacity on edge e and y_e to be the number of additional facilities to be installed on edge e . Then the total capacity on edge e is $u_e + Cy_e$. We

use Theorem 5.5 to formulate the CSP as the following mathematical program:

$$\begin{aligned}
(\text{CSP}) \quad & \text{Minimize} \quad \sum_{e \in E} c_e \cdot y_e \\
& \sum_{e \in E} \ell_e(u_e + C y_e) \geq \max_{d: V d \leq w} \left(\sum_{k \in K} d_k \cdot \text{dist}_\ell(o(k), \delta(k)) \right) \quad \forall \ell : E \rightarrow \mathbb{Z}_+ \quad (5.5a) \\
& y_e \in \mathbb{Z}_+, \quad \forall e \in E.
\end{aligned}$$

Clearly, the number of possible length functions are infinite. So, as stated, this integer program is not useful for solving the CSP. We first examine the complexity of separating the constraints (5.5a). Then, we consider special cases for which a ‘small’ subset of these constraints are sufficient to ensure feasibility.

5.3.2 The Separation Problem

We consider the separation problem associated with the necessary and sufficient condition we established for the CSP. That is, given a vector of arc capacities u , we must either show that the capacities satisfy all of the constraints (5.4) or present a constraint that is violated. We show that this problem is NP-Hard.

First we identify an important subset of these constraints that allows a physical interpretation. Given a subset S of the node set V , let \bar{S} be the set of nodes not in S . A cut, denoted by (S, \bar{S}) , is the set of edges that have one of their endpoints in S and the other in \bar{S} . Consider a length function ℓ^S that is 1 for all the edges in a given cut (S, \bar{S}) and 0 otherwise. For this length function, the condition (5.4) becomes the “cut constraints”:

$$\sum_{\langle i, j \rangle \in E: i \in S, j \notin S} u_e \geq \max_{d: V d \leq w} \left(\sum_{i \in S, j \notin S} d_{ij} \right). \quad (5.6)$$

That is, the capacity across the cut must be at least the maximum total demand across the cut. For convenience, we have slightly changed notation by denoting d_k with $o(k) = i$ and $\delta(k) = j$ by d_{ij} . There is one such constraint for each cut. Cut constraints are a subset of the feasibility condition (5.4). Therefore, cut constraints are necessary but not sufficient for the feasibility of the capacity sharing problem.

We now show that separating constraints 5.5a is NP-Hard. Since the cut constraints are a subset of the feasibility condition, it will suffice if we show that separating cut constraints is NP-Hard.

Theorem 5.6. *Separating cut constraints for the capacity sharing problem is strongly NP Hard.*

Proof. We provide a polynomial reduction from the strongly NP-Hard Set Cover problem. In the Set Cover problem, we are given a set U of s elements and a collection C_1, C_2, \dots, C_t of subsets of U whose union is U . Each subset C_j has a cost c_j . We seek to identify a minimum cost collection of subsets so that each element of U is contained in at least one of the selected subsets.

Given an instance of Set Cover, we create a network as follows: We create a node for each element u_i of the set U , and a node for each subset C_j . We also create an additional node 0. The only edges are those between the node 0 and every other node. The capacities on the edges are as shown in Figure 5-1. We choose $M > n \sum_{1 \leq j \leq t} c_j$.

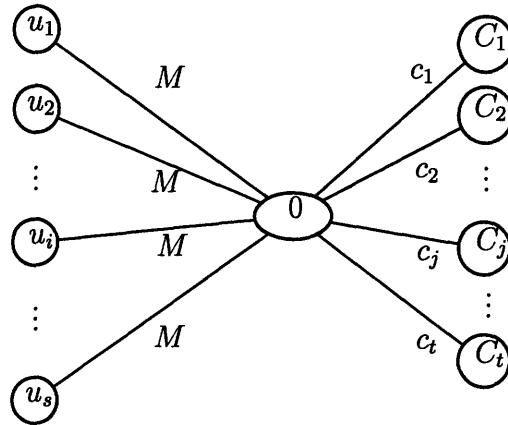


Figure 5-1: Reduction from Set Cover Problem

We impose a demand between an element u_i and a subset C_j if C_j contains u_i . The demand variation is the following polyhedron: $\{d : \sum_{j:C_j \ni u_i} d_{ij} \leq L, 1 \leq i \leq s\}$. We choose L to be large, $L > \sum_{1 \leq j \leq t} c_j$.

Assume that $0 \in S$. Therefore, the capacity of the cut S is given by $\sum_{i:u_i \notin S} M + \sum_{j:C_j \notin S} c_j$. We claim that the given instance of Set Cover has a cover with cost at

most γ if and only if the network has a cut S , and for some demand d in the given variation $\sum_{i:u_i \notin S} M + \sum_{j:C_j \notin S} c_j - (\sum_{i \in S, j \notin S} d_{ij} + \sum_{i \notin S, j \in S} d_{ij})$ is at most $\gamma - nL$. This result directly implies the NP-Hardness of the separation problem.

Given a cover with cost γ , we choose all nodes to be in the set S except the nodes corresponding to the subsets in the cover. For the cut S , the capacity across the cut is just the cost of the subsets in the cover, that is, γ . For every element u_i in U , we choose a subset C_j in the cover containing u_i and set d_{ij} to be L . We set d_{ij} to be zero for all other demands. The demand across the cut in this case is nL , which yields the ‘only if’ part of the claim.

Now suppose that the difference of the total demand across some cut from the capacity of the cut is $\gamma - nL$, for some $\gamma \leq \sum_{1 \leq j \leq t} c_j$. Since we chose M to be large, none of the arcs with cost M can be part of the cut. So, all the element nodes must be in S . The capacity of the cut S is at most $\sum_{1 \leq j \leq t} c_j$. Since $\gamma < L$, the demand across the cut must be nL . Since each element node has a total demand of at most K , we conclude that the nodes that are not in S is a cover of U . Also, the cost of this cover is γ , establishing the ‘if’ part of the claim. \square

5.3.3 When Cut Constraints Are Sufficient

The general necessary and sufficient condition 5.4 for a edge capacity vector to be feasible for all demands in a variation set contains an infinite number of inequalities, and therefore is not useful for designing a solution procedure for the CSP. A model restricted only to cut constraints, which are finite but still exponential in number, will be more manageable. Although we showed that separating cut constraints is NP-Hard, cut constraints have been successfully used in a variety of network design and capacity expansion problems, and we can heuristically identify violated cuts. If for a network, cut conditions are sufficient to ensure that a given capacity vector is feasible for the CSP, we can use this restricted model, and develop a heuristic procedure to solve it. We seek the conditions under which the cut constraints are sufficient for the CSP.

Definition 5.1. *The demand graph $H = (T, R)$ associated with a capacity expansion problem is the undirected graph with node set T and edge set R for which*

- *T contains all the nodes that are either the origin or the destination of some nonzero demand*
- *R contains an edge between the origin and destination of every nonzero demand.*

For example, if there is demand between every pair of nodes in the network, the demand graph is a complete graph on the nodes of the network. If all the demands originate at the same node s , then the demand graph is a star network with node s as the hub.

The following theorem, due to Papernov [45] (see Schrijver [49]), states the conditions under which cut constraints are sufficient for feasible multicommodity flow to exist for a given demand. The condition is independent of the network G , and depends only on the demand graph H .

Theorem 5.7. *(Papernov [45]) For a network G with a demand graph H , the cut conditions imply the existence of a multicommodity flow if and only if $H = K_4$, or $H = C_5$, or H is the union of two stars.*

Here, K_4 is the complete graph on 4 nodes and C_5 is the cycle with 5 nodes. Figure 5-2 illustrates the three demand graphs of Theorem 5.7.

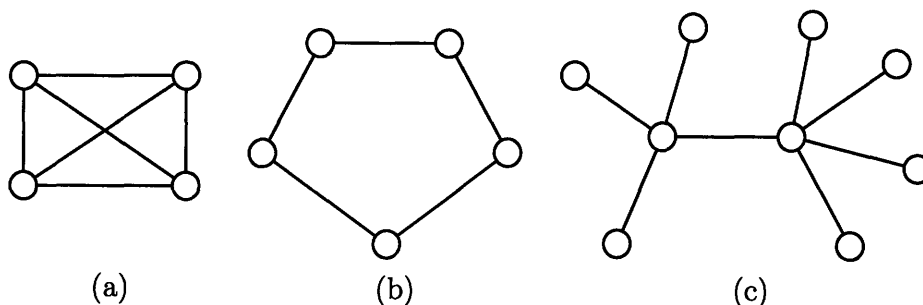


Figure 5-2: Demand graphs for which cut conditions are sufficient. (a) K_4 (b) C_5 (c) Union of two stars.

This condition obviously extends to the Capacity Sharing Problem. The righthand side in the cut conditions for a multicommodity flow problem is the total demand

across the cut. For the Capacity Sharing Problem the righthand side is the maximum total demand across the cut among all the demands in the given demand variation set.

Theorem 5.7 suggests the following approach to solve the CSP. Partition the set of all commodities K (that is, edges in the demand graph) into subsets K^1, K^2, \dots, K^p , each of which forms one of the three demand graphs in Theorem 5.7. We then introduce variables z_e^ℓ for each subset K^ℓ and every edge in E for the amount of capacity ‘reserved’ on the edge e for transporting the commodities in the set K^ℓ . We can now approximate the CSP as the following alternate integer program:

$$\begin{aligned} \text{(CSP)} \quad \text{Minimize} \quad & \sum_{e \in E} c_e \cdot y_e \\ & \sum_{1 \leq \ell \leq p} z_e^\ell \leq u_e + C y_e, \quad \forall e \in E \quad (5.7a) \end{aligned}$$

$$\sum_{e=(i,j) \in E: i \in S, j \notin S} z_e^\ell \geq \max_{d: V d \leq w} \left(\sum_{k \in K^\ell, o(k) \in S, \delta(k) \notin S} d_k \right), \quad \forall S \subset V, 1 \leq \ell \leq p \quad (5.7b)$$

$$z_e^\ell \geq 0, \quad \forall e \in E, 1 \leq \ell \leq p$$

$$y_e \in \mathbb{Z}_+, \quad \forall e \in E.$$

This integer program is an approximation of the CSP and not always an exact formulation because when we partition the set of commodities into subsets, two commodities in different subsets can no longer share capacity (since we reserve capacity for each subset separately). We now examine the tightness of this approximation. Given a polyhedral demand variation set U , and a subset of commodities K^ℓ , let $\pi_U(K^\ell)$ be the projection of U to the commodities in K^ℓ .

Theorem 5.8. *The formulation (5.7) is equivalent to the Capacity Sharing Problem with the demand variation set $\pi_U(K^1) \times \pi_U(K^2) \times \dots \times \pi_U(K^p)$.*

Proof. Consider the cut constraints associated with the commodity subset K^ℓ . For any cut S , the righthand side of the cut constraint is $\max_{d: V d \leq w} \left(\sum_{k \in K^\ell, o(k) \in S, \delta(k) \notin S} d_k \right)$.

We are interested only in the commodities across the cut that are in the set K^ℓ .

Equivalently, we can write this cut constraint as

$$\sum_{e=(i,j) \in E: i \in S, j \notin S} z_e^\ell \geq \max_{d \in \pi_U(K^\ell)} \left(\sum_{k \in K^\ell, o(k) \in S, \delta(k) \notin S} d_k \right). \quad (5.8)$$

Note that demand for a commodity k occurs only in the cut constraints corresponding to the subset K^ℓ that contains k . Therefore, in the integer program (5.7), if two commodities k_1 and k_2 are in different subsets, they vary independent of each other. Therefore, the set of all demands that will be feasible for the capacity plan produced by the integer program (5.7) is the polytope $\pi_U(K^1) \times \pi_U(K^2) \times \cdots \times \pi_U(K^p)$. \square

So, in effect, the cut constraint approximation to the CSP changes the demand variation set from U to $\pi_U(K^1) \times \pi_U(K^2) \times \cdots \times \pi_U(K^p)$. Since $\pi_U(K^1) \times \pi_U(K^2) \times \cdots \times \pi_U(K^p)$ contains U , the integer program (5.5a) is an approximation to the CSP.

Figure 5-3 illustrates the change in demand variation set on a three commodity CSP. The polytope on the left hand side is the initial demand variation set U . When we partition the commodity set $\{d_1, d_2, d_3\}$ into two subsets $\{d_1, d_2\}$ and $\{d_3\}$, and reserve capacity for each subset separately, the effective demand variation set changes to the figure in the right hand side.

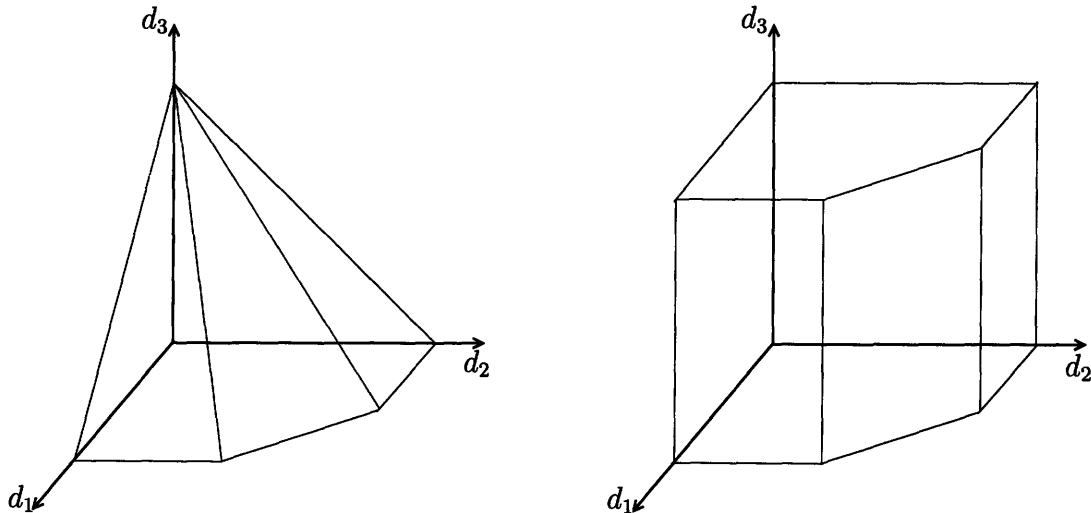


Figure 5-3: The original demand variation and the approximation by the partition $\{d_1, d_2\}, \{d_3\}$.

We now establish a worst case bound for this approximation.

Theorem 5.9. *Let P be a Capacity Sharing Problem with demand variation set U , and commodity set K . Given a partition of the commodity set K into subsets K^1, K^2, \dots, K^p , let P' be the integer programming problem (5.7) with these subsets. Then the optimal cost of P' is at most p times the optimal cost of P .*

Proof. Let Z and Z' be the optimal cost to the problems P and P' . Now consider the Capacity Sharing Problem P^i defined on the same network but containing demands only from the subset K^i . In P^i , all other demands are zero. Let Z^i be the optimal cost of P^i . Clearly, $Z^i \leq Z$. Also, since the number of new facilities y_e on every edge e is required to be integer, we have

$$\sum_{1 \leq i \leq p} Z^i \geq Z'.$$

Therefore, $Z' \leq pZ$. □

Since the solution for P' is feasible for P , if we can solve P' , we have a p -approximation algorithm for the CSP. Since the approximation ratio depends on the number of partitions, we obtain a tighter approximation guarantee by partitioning the commodities into smaller subsets (ensuring that the demand graph for each subset is K_4 , C_5 or the union of two stars). The approximation ratio we established is independent of the demand variation set U . We will use this idea of partitioning the commodity set to design a heuristic procedure in Section 5.4.3.

5.4 Solving The Capacity Sharing Problem

We consider a single facility Capacity Sharing Problem in an arbitrary network $G = (V, E)$. Assume that the capacity cost is edge-dependent and linear in the number of facilities. Finally, the demand variation set U is given as a polytope $Vd \leq w$. We present three heuristic approaches for solving the CSP. The first two approaches are known algorithms that could be used to approximate the CSP, and we establish some bounds for their performance. We develop a third heuristic, based on a cutting plane approach, that addresses demand variation directly.

5.4.1 Single Demand Optimization

In this model, for each commodity k , we evaluate the maximum demand $\hat{d}_k = \max\{d_k : Vd \leq w, d \geq 0\}$. We then use the maximum demand vector \hat{d} to solve a single demand forecast capacity expansion problem. From Observation 5.1, we know that the resulting capacity vector will be feasible for all demands in the polyhedral variation set U_P . Figure 5-4 illustrates Single Demand Optimization for a Capacity Expansion Problem with two commodities and a polyhedral variation set.

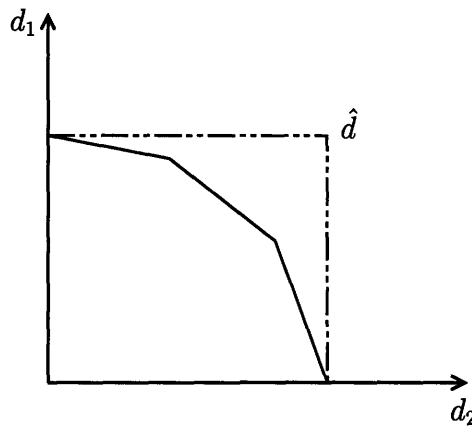


Figure 5-4: The Single Stage Model

In the figure, d_1 and d_2 are the demands of the two commodities. The variation set is the polytope shown in the figure by solid lines. Solving the capacity expansion problem for the demand \hat{d} gives a capacity plan that is feasible for all the demand vectors in the square (shown in Figure 5-4 by dashed lines), and therefore feasible to all points in the variation polytope too.

The single demand optimization essentially ‘relaxes’ the demand variation polytope to a box (or a cube). So we can view this method as an approximation to the Capacity Sharing problem. This method is the current state of the art of telecommunication network planning. The difference between the demand variation polytope and the box polytope provides an opportunity for cost savings.

Let k be the number of commodities in the CSP, and let K^i be the commodity subset containing only commodity i . Then the box polytope $\{d \in \mathbb{R}^k : 0 \leq d \leq \hat{d}\}$, which is the effective demand variation for the Single Demand Optimization, can

be expressed as $\pi_U(K^1) \times \pi_U(K^2) \times \cdots \times \pi_U(K^k)$. That is, the Single Demand Optimization is equivalent to the cut constraint approximation with the partition K^1, K^2, \dots, K^k .

Observation 5.10. *If k is the number of commodities in a Capacity Sharing Problem, then the optimal cost of the Single Demand Optimization problem is at most a factor of k from the optimal cost of the CSP.*

Proof. Follows from Theorem 5.9 and the observation we made above. □

Again, we note that the performance guarantee established above is independent of the demand variation set U .

5.4.2 Fractional Flow Variables

The single demand optimization problem solves the capacity expansion problem for the component wise maximum demand vector in the given demand variation set. Another way to state the single demand optimization problem in the presence of demand variation is to require that the capacity variables and a single vector of flow variables be simultaneously chosen. The flow vector we choose must satisfy all the demands in the variation set, so the flow is feasible to the component wise maximum demand.

A slightly better approach than single demand optimization is to define fractional flow variables λ_{ij}^k that denotes the fraction of the demand d_k that flows on the arc (i, j) . We now require that the fractional flow variables be chosen along with the capacity variables. The actual flow on the arc depends on the demand vector in the variation that is currently active. We present a formulation for the Capacity Expansion Problem with demand variation U with fractional flow variables.

$$\begin{aligned}
\text{(CEP)} \quad & \text{Minimize} \quad \sum_{e \in E} c_e \cdot y_e \\
& \sum_{(i,j) \in A} \lambda_{ij}^k - \sum_{(j,i) \in A} \lambda_{ji}^k = \begin{cases} 1 & \text{if } i = o(k) \\ -1 & \text{if } i = \delta(k) \\ 0 & \text{otherwise.} \end{cases} \quad \forall i \in V, \forall k \in K \quad (5.9a) \\
& \sum_{k \in K} d_k (\lambda_{ij}^k + \lambda_{ji}^k) \leq u_e + C \cdot y_e, \quad \forall e = \langle i, j \rangle \in E, \forall d \in U \quad (5.9b) \\
& y_e \in \mathbb{Z}_+, \quad \forall e \in E \\
& 0 \leq \lambda_{ij}^k \leq 1, \quad \forall e = \langle i, j \rangle \in E, \forall k \in K.
\end{aligned}$$

We highlight the connection of the fractional flow variables CEP to a result for Adjustable Robust Optimization by Ben-Tal et al. [13]. In Adjustable Robust Optimization, the problem has two sets of variables. One set of variables are to be chosen under uncertainty, and the other set, called ‘adjustable variables,’ can be chosen once the uncertainty is realized. The Capacity Sharing Problem can be viewed as an instance of Adjustable Robust Optimization: in the formulation (5.1), the arc capacity variables y are ‘fixed’, while the flow variables f are adjustable and can be chosen after a demand from the variation set U is realized. Ben-Tal et al. show that the Adjustable Robust Linear Program is computationally tractable when the adjustable variables can be expressed as an affine function of the realized uncertain parameters. The formulation (5.9) with fractional flow variables is an approximation to the CSP where we restrict the flow variables to be linearly related to the realized demand d , i.e., $f_{ij}^k = d_k \lambda_{ij}^k$.

Mudchanatongsuk et al. [42] consider this problem, and using duality theory, propose an equivalent integer program with a polynomial number of variables for the problem (5.9). The authors also observe that this approach produces conservative solutions that do not realize much of the possible savings due to capacity sharing.

Since this approach produces a solution at least as good as the single stage optimization, we conclude that for a CSP with k commodities, the optimal cost of the

fractional flow problem is at most k times the optimal cost of the CSP.

5.4.3 A Cutting Plane Heuristic By Partitioning Commodities

In Section 5.3.3, we proposed an idea for a heuristic for CSP: Partition the commodity set K into subsets K^1, K^2, \dots, K^p such that the demand graph for each K^ℓ satisfies Papernov's condition. We then heuristically solve the integer program (5.7) for the partition K^1, K^2, \dots, K^p using a cutting plane approach.

We start with a number of cuts for each subset. At each iteration, we solve a master problem: the integer program (5.7) in which cut constraints are included only for the current cuts. Using the solution values of the capacity variables z_e^ℓ in the solution to the master problem, we solve a separation problem (exactly or heuristically) for each subset K^ℓ . The separation problem is to identify a violated cut constraint if one exists, and declare that none exists otherwise. When a violated cut constraint is identified, we add the corresponding cut(s) to the master problem and continue. The heuristic stops when there are no violated cuts.

We have not yet stated how we will partition the commodity set K . Theorem 5.9 suggests that it might be better to partition K into a small number of subsets. However, we would also like to choose the partition so that the resulting separation problem for each subset is reasonably simple. We will partition K into $|V|$ subsets, one for each node in the network, with the subset for node i containing all the demands that originate at node i . The demand graph for each of these subsets is a star network and hence satisfies Papernov's condition.

The performance of the heuristic depends also on how well we can solve the separation problem for the commodity subsets we have created. We model this separation problem in Section 5.4.3. While the problem is still NP-Hard, we found that it is not difficult to solve in practice for real sized instances. We also propose a heuristic for the separation problem that can be used to speed up the cutting plane procedure.

The heuristic procedure we propose for the (capacity sharing version of the) Ca-

capacity Expansion Problem could also be applied for the more general Capacitated Network Design Problem. In each iteration, we would solve a Network Design master problem with the current set of cut constraints instead of a Capacity Expansion master problem. The practical utility of the procedure would depend on how fast we could solve the master problem with a reasonable number of cuts.

The Single Source Separation Problem

Since the cutting plane heuristic we will use partitions the commodity set by source node, the separation problem for each source node s assumes the following form. Given a vector of arc capacities z^v , we want to find the cut whose demand from node s to nodes on the other side of the cut most exceeds the capacity across the cut. We call this problem the single source separation problem.

Since all the commodities originate at the source node s , the maximum demand across a cut S is $\max_{d \in U} \sum_{i \notin S} d_{si}$. We assume that the source node s is always in the selected cut S . Let x_i be a decision variable that is 1 when the node i is not in the selected cut S . The following integer program solves the separation problem:

$$\text{Minimize } \sum_{e \in E} z_e \cdot y_e - \sum_{i \in V \setminus \{s\}} d_{si}$$

$$y_e \geq x_i - x_j, \quad \forall e = \langle i, j \rangle \in E \quad (5.10a)$$

$$y_e \geq x_j - x_i, \quad \forall e = \langle i, j \rangle \in E \quad (5.10b)$$

$$x_s = 0 \quad (5.10c)$$

$$\sum d \leq w \quad (5.10d)$$

$$d_{si} \leq Mx_i \quad \forall i \in V \setminus \{s\} \quad (5.10e)$$

$$x_i \in \{0, 1\}, \quad \forall i \in V$$

$$y_e \in \{0, 1\}, \quad \forall e \in E$$

$$d_{ij} \geq 0, \quad \forall i, j \in V.$$

The constraints (5.10a) and (5.10b) set y_e to 1 if the edge e crosses the chosen cut. The

constraint (5.10c) forces the source node to be in the selected cut. Constraints (5.10d) are the demand variation constraints and constraints (5.10e) ensure that the demand of a commodity is positive only if the destination of that demand is on the other side of the cut from node s .

We now show that even the single source separation problem is NP-Hard.

Theorem 5.11. *The single source separation problem is NP-Hard*

Proof. We present a polynomial time reduction from the weakly NP-Hard minimum knapsack problem. In the minimum knapsack problem, we are given n items with sizes w_i and penalties p_i . The objective is to choose a set of items of total size at least W to exclude from the knapsack that minimizes the penalty incurred by the chosen items.

Given an instance of the minimum knapsack problem with n items, we create a network with $n + 1$ nodes: one node for each item and a special source node s . There is an edge from the source node s to each of the other nodes. The capacity on the edge $\langle s, i \rangle$ is p_i . Let $P > \sum_{1 \leq i \leq n} p_i$. The demand variation for the separation problem is given by

$$0 \leq d_{si} \leq w_i P, \quad \sum_{1 \leq i \leq n} d_{si} \leq WP.$$

We show that there is a solution to the separation problem with cost at most $\gamma - WP$, if and only if the minimum knapsack problem has a solution with cost at most γ . Given a solution to the separation problem with cost $\gamma - WP$, we conclude that $\sum_{1 \leq i \leq n} d_{si} = WP$. Since d_{si} can be positive only when x_i is 1, $\sum_{1 \leq i \leq n} w_i x_i \geq WP$. Therefore, we can choose all the items with $x_i = 1$ to exclude, and the total penalty of these items is just the capacity across the cut, which is $(\gamma - WP) + WP = \gamma$.

On the other hand, if we are given a solution with cost γ to the minimum knapsack problem, we can choose all the items that are not excluded along with the source node s to be a part of the cut S . It is easy to verify that the difference between total capacity across the cut S (γ) and the maximum demand across the cut (WP) is $\gamma - WP$.

This implies that the separation problem is NP-Hard. □

A Heuristic For The Separation Problem

In our computational experiments, we observed that the single source separation problem, when solved as an integer program, usually ran within a few seconds even on networks with 50 nodes. Solving an integer program during each iteration of the heuristic might not be possible for larger networks. Also, a heuristic can be useful if it can find more than one violated cuts in each iteration (even if none of them is the most violated cut) as this might speed up the heuristic by reducing the number of iterations. We present one such heuristic for the single source separation problem.

We observe that if we fix the x variables, the integer program reduces to a linear program in the demand variables, and can be solved easily. On the other hand, as we show, if we fix the demand variables, the problem can be solved as a single minimum cut problem. Given fixed demand d_{si} , we reformulate the separation problem as follows:

$$\text{Minimize } \sum_{e \in E} z_e \cdot y_e - \sum_{i \in V \setminus \{s\}} d_{si} x_i$$

$$y_e \geq x_i - x_j, \quad \forall e = \langle i, j \rangle \in E \quad (5.11a)$$

$$y_e \geq x_j - x_i, \quad \forall e = \langle i, j \rangle \in E \quad (5.11b)$$

$$x_s = 0 \quad (5.11c)$$

$$x_i \in \{0, 1\}, \quad \forall i \in V$$

$$y_e \in \{0, 1\}, \quad \forall e \in E$$

$$(5.11d)$$

We can rewrite the objective function as

$$\sum_{e \in E} z_e \cdot y_e + \sum_{i \in V \setminus \{s\}} d_{si}(1 - x_i) - \sum_{i \in V \setminus \{s\}} d_{si}.$$

Since the term $\sum_{i \in V \setminus \{s\}} d_{si}$ is a constant, we can ignore it. We now introduce a new node t and create edges with capacity d_{si} from node t to every node i . The new objective function is just the capacity of the chosen cut S in this network. Therefore,

we can solve the separation problem for a fixed demand as a minimum cut problem.

We suggest the following heuristic for the Single Source Separation Problem:

1. Find demand d maximizing $\sum_{i \in V \setminus \{s\}} d_{si}$.
2. Fixing d , solve problem (5.11).
3. Using the cut S obtained in Step 2, find demand d maximizing $\sum_{i \notin S} d_{si}$.
4. Repeat steps 2-3 until no improvement is possible.

At the end of step 3, we can check if the capacity of the cut S is more than the maximum demand across the cut. If it is, we can add the cut to the master problem. We can decide to either terminate the heuristic at this point, or continue to look for more violated cuts.

An Improvement For The Cutting Plane Heuristic

In our solution procedure for the capacity expansion problem, we solve the master problem with the current set of cuts. We use the resulting arc capacities and solve a separation problem for each source node. We terminate the procedure when the separation problem for every source has zero objective. However, in our computational experiments, we observed that for many instances of the capacity expansion problem, the objective of the separation problem for certain switches reaches a small value relatively soon. Then, the cut generation procedure takes several more iterations until the optimal cost of the separation problem becomes nonnegative. We propose an approximation approach that will help reduce the time spent on iterations after the separation objective has reached a small value.

Observation 5.12. *Let the optimal solution to the current separation problem be $-\epsilon$ for some $\epsilon > 0$. If we add ϵ to the capacity of every arc in a spanning tree of the network, the resulting set of capacities will be feasible for all the cut constraints.*

Proof. If the optimal cost of the separation problem is $-\epsilon$, the capacity across all cuts in the network is at least the demand across the cut less ϵ . Therefore, if we add

ϵ to the capacity of all cuts, the separation problem with the new capacities will have nonnegative objective. If we add ϵ to the arcs of a spanning tree, the capacity of every cut in the network increases by at least ϵ . \square

For each source node, we stop adding cuts for a source node when the objective of its separation problem reaches a prespecified (negative) number. Let the optimal cost of the separation problem for source i be $-\epsilon_i$ when we stop adding cuts for the source node. After the heuristic procedure stops (that is, when we stop adding cuts for all the source nodes), we add $\sum_{i \in V} \epsilon_i$ to the edges of a spanning tree of G to obtain a feasible capacity vector for the CSP.

To do this with minimal increase in cost, we could compute, for each edge, the cost of adding $\sum_{i \in V} \epsilon_i$ units of capacity. For the single facility case, it could be that an edge has enough spare capacity that this capacity addition can be accomplished at no cost. We compute a Minimum Spanning Tree with respect to these costs, and add $\sum_{i \in V} \epsilon_i$ units of capacity on the edges of this tree. In our computational experiments, we observed that this procedure improved the run time of the heuristic with a very modest increase in cost.

Chapter 6

Computational Experiments

We tested the cutting plane heuristic proposed in Chapter 5 computationally on real and random telecommunication networks. We used network data from a long distance service provider. For demand data, we created one problem set based on the single point demand forecast of a long distance provider and another problem set based on random demand variation. We also generated problems on random networks comparable to real telecommunication networks. We implemented a version of the cutting plane heuristic, and tested it on all the data sets. Our results show that capacity sharing can reduce the network planning budget significantly with a savings of 10-30% for real networks. The heuristic we propose performs well, producing solutions within 10% of optimality for most of the instances tested.

6.1 Algorithm Summary and Implementation

We tested our heuristic on several Network Loading Problems with Capacity Sharing. That is, in all our instances, the network has no initial capacity installed on the arcs. We assumed a single facility type, which was a type DS3 (equivalent to 672 simultaneous calls) for all problem sets.

Our algorithm solves the linear relaxation of the network loading problem with capacity sharing, and rounds the solution up to obtain integer number of facilities on each edge. We partitioned the commodities by source and introduced an edge

capacity variable for every commodity subset. In this case, since the initial capacities are zero, the linear relaxation of the capacity sharing problem decomposes by source. We solve for the capacity reserved for each source separately.

We start with a set of initial cuts. This set consists of all singleton and two node cutsets. At each iteration, we solve the separation problem as an integer program. If the optimal cost of the separation problem is negative, we add this cut to the master problem and continue.

We implemented the improvement procedure for early termination that we discussed in Section 5.4.3. As before, let C be the facility capacity. For a n node network, whenever the optimal cost of the separation problem exceeded $-2 * C/n$, we terminated the master problem, and added the optimal cost to the edge capacities of a minimum spanning tree at the end. We observed that this approach resulted in slightly faster running times.

Finally, after solving n such problems, for each edge we added the capacities reserved for each subproblem, and rounded it up to the next integer.

We conducted the computations on a Pentium IV 3.2 GHz machine, with 2 GB Random Access Memory and running Linux Operating System. We implemented the algorithm using Optimization Programming Language (OPL) 5.1 running CPLEX 10.0. Using OPL causes some additional overhead in the runtime, and so it possible that implementing the algorithm directly using CPLEX libraries would improve the runtime.

6.2 Test Problems

We used data from three real long distance networks. Table 6.1 specifies their dimension. We could not obtain demand variation data for the three networks from the long distance service provider. Instead, we made reasonable assumptions to generate demand variations for solving the Capacity Sharing Problem from single point peak demand forecasts for the three networks.

For the first real data set, we made the following assumptions regarding the de-

mand variation that we use to solve the Capacity Sharing Problem. We assumed that 70% of the single point demand forecast occurs at all times. That is no sharing is possible for this part of the demand. The remaining 30% constitutes the peak demand, and these peaks do not all occur at the same time. So we added a constraint restricting the total peak demand, that is the sum of all the demands above the 70% base demand, is never more than one fifth of the maximum possible peak demand (when all the demands are at their maximum possible value). That is, we assume that the total demand in the network for all the O-D pairs is not more than $70\% + (1/5)30\% = 76\%$ of the sum of the individual peak demands. This, we believe, is a reasonable model consistent with real telecommunication traffic data.

For the second real data set, we used real network data, but we generated the demand variation set randomly. We generated 10 constraints for each source node for the demand variation. For each constraint, we specify a range $[0, K]$ from which the coefficients for each O-D demand would be randomly selected. For four out of the ten constraints, we set $K = 1$ so that all the coefficients for these constraints are 0-1. For the other six constraints, we set K to different values between 10 and 50. The righthand side of the constraints were selected depending on the value K for the constraint so that the demands in the variation would roughly correlate with the real demands in the network.

In addition to real networks, we generated random instances with three different network sizes: 10, 20 and 30 nodes. We built and used a random network generator that for a given number of input nodes and average node degree, creates a random network. It will first generate a random spanning tree and chooses the remaining arcs with equal probability that depends on the required average degree. While the expected average degree of the created network is equal to the required average degree,

Instance	R1	R2	R3
$ N $	27	29	50
$ E $	330	262	749
Average Node Degree	24.4	18.1	30.0

Table 6.1: Real problem instances

a network generated by the procedure could have higher or lower average degree. We randomly sampled edge costs from a range that is consistent with real world data. Finally, for demand variation, we generated a 10 constraint polyhedron for each source node in the same way as we did for the real networks with random demand data.

6.3 Computational Results

In our computational results for testing our heuristic, after running each instance, we calculated the percentage gap of the solution we obtained from a lower bound for the Capacity Sharing Problem. The heuristic solves the linear relaxation of the Capacity Sharing Problem and rounds the solution. There are two contributors to this gap. One gap arises if we terminate when the separation problem still has negative objective (due to the early termination procedure). The other contribution to the gap is caused by rounding the solution in the end to obtain an integer number of facilities.

We also compare our solution to single demand optimization. That is, we calculate the maximum demand for each destination and plan the network to support all these demands simultaneously. Since, this approach is roughly indicative of current practice, the difference in cost indicates the amount of savings due to capacity sharing.

We sound a note of caution about savings figures especially for instances for which the demand variation has been generated randomly. The polyhedral demand variations that we generate randomly might not portray a telecommunications network's demand pattern accurately and therefore, the savings that we report in our computational results must not be interpreted as a percentage of a company's total budget. We offer the following alternate interpretation of the savings figures presented in our results for those problem instances with random demand variation. A telecommunication network's demand has a more or less steady base demand (the amount of capacity utilized on most days), and a peak demand, that occurs occasionally. We believe that a random demand variation more accurately models the excess demand

above the median. Therefore, the savings that we report should be considered percentages of the part of the budget that is allocated to meeting the excess demands. As an example, if the median demands are on average 70% of the peak demand, our reported savings of 50% would translate to roughly 15% of the budget.

Table 6.2 shows the results for three real instances with demand data generated based on real single point demand forecasts.

Instance	Gap	Time	I. Cuts	T. Cuts	Savings
R1	11.76%	57s	378	378.11	13.4%
R2	9.92%	51s	435	437.07	12.2%
R3	9.92%	20m 39s	1275	1276.16	28.7%

Table 6.2: Computational results for real problem instances

For this and other tables, ‘Gap’ is the percentage difference between the heuristic solution and a lower bound for the capacity sharing problem. ‘Time’ is the total runtime for the procedure. We report time in hour (h), minute (m), second (s) format. ‘I. Cuts’ is the number of initial cuts added per source, and ‘T. Cuts’ is the average total number of cuts added per source. Finally, ‘Savings’ is the percentage reduction in cost due to the heuristic when compared to capacity plan produced by the single demand optimization.

For all the three instances, the heuristic produced solution with a gap of less than 3%. The initial cuts alone were enough to produce solutions that meet the termination criteria for almost all the source node subproblems in each of the three cases. We added less than 3 cuts per source node for all three problems. Finally, when compared to single demand optimization, the heuristic was able to save between 10 and 30 percent.

Table 6.3 presents the results for the second data set, which used real network data and random demand data. Since the demand variation set is more complicated than in the previous case, the method added about 20 cuts (Total Cuts - Initial Cuts) per source node for each of the three problems. In terms of run time, all three problems took much longer with the problem R3 taking more than a day to complete. We observed that the bottleneck was not solving the master or the separation problem,

but generating the master problem during every iteration. Since the method required only a few iterations, we believe that a better implementation (possibly directly using CPLEX libraries) would significantly improve the speed of the heuristic.

Instance	Gap	Time	I. Cuts	T. Cuts	Savings
R1	4.55%	1h 41m	378	399.60	61.7%
R2	4.52%	1h 23m	435	456.90	58.1%
R3	0.64%	25h 09m	1275	1298.22	67.2%

Table 6.3: Computational results for real networks with random demand

For this data set, the heuristic cost is 60% less than the single demand optimization cost in all three cases.

No.	$ E $	Gap	Time	I. Cuts	T. Cuts	Savings
1.	20	9.47%	11.3s	55	59.1	52.2%
2.	26	18.74%	18.0s	55	61.3	51.4%
3.	28	16.70%	11.0s	55	59.0	47.5%
4.	23	10.97%	10.7s	55	58.7	55.3%
5.	23	9.80%	8.5s	55	57.4	54.1%
6.	19	10.33%	10.3s	55	59.0	51.5%
7.	24	9.77%	12.5s	55	59.6	47.4%
8.	23	7.59%	9.5s	55	58.2	52.7%
9.	19	10.21%	9.9s	55	58.8	52.0%
10.	24	11.92%	14.5s	55	60.5	53.9%
Avg.	22.9	11.55%	11.6s	55	59.2	51.8%

Table 6.4: Computational results for random 10 node networks

Tables 6.4, 6.5, and 6.6 present our final results for random instances. We tested 10 instances each with 10 and 20 nodes, and 6 instances with 30 nodes. In all instances, the average gap was less than 12%. The gap decreased with the instance size. With random variation in demand, the heuristic saved around 50% for every instance tested. As the problem size increases, the number of cuts required per source node also increases. Since the complexity of solving each iteration also increases with problem size, these two factors together cause a nonlinear increase in run time. In spite of that, we believe the capacity sharing model as well as the cutting plane heuristic

is still valuable, and can help telecommunication planners significantly reduce their investments in network capacity.

<i>No.</i>	$ E $	Gap	Time	I. Cuts	T. Cuts	Savings
1.	84	6.29%	36m 09s	210	300.25	63.4%
2.	101	7.14%	36m 06s	210	294.65	64.6%
3.	90	8.06%	31m 06s	210	290.90	64.3%
4.	93	6.92%	33m 33s	210	295.65	64.0%
5.	88	6.66%	29m 09s	210	289.65	65.3%
6.	89	6.80%	35m 31s	210	297.50	64.2%
7.	87	6.96%	28m 30s	210	287.60	63.7%
8.	96	7.46%	37m 07s	210	299.45	64.1%
9.	75	7.09%	24m 09s	210	287.95	64.4%
10.	96	8.23%	31m 06s	210	289.65	64.7%
Avg.	89.9	7.16%	32m 14s	210	293.33	64.3%

Table 6.5: Computational results for random 20 node networks

<i>No.</i>	$ E $	Gap	Time	I. Cuts	T. Cuts	Savings
1.	150	2.40%	6h 08m	465	673.00	70.5%
2.	151	2.74%	6h 08m	465	673.73	71.1%
3.	147	2.85%	5h 56m	465	641.43	71.0%
4.	153	2.72%	5h 47m	465	667.43	71.8%
5.	140	2.78%	5h 29s	465	667.13	70.7%
6.	158	2.46%	5h 43s	465	677.13	71.7%
Avg.	149.8	2.66%	5h 52m	465	671.64	71.1%

Table 6.6: Computational results for random 30 node networks

Chapter 7

Conclusions

We studied three problems faced by contemporary telecommunications network planners. The first two problems are motivated by real problems faced by a major long distance carrier. Both problems address capacity expansion problems on hybrid networks that are prevalent today. The third problem, the Capacity Sharing Problem, uses the fact that peak demands might occur at different times in deciding how much capacity to add to the network. Planning the network using Capacity Sharing could potentially lead to significant savings compared to current planning models that do not take advantage of efficient call routing technologies. We summarize our contribution to the literature on telecommunication network capacity planning as follows.

Capacity Expansion Problem (CEP)

- We presented a $(2+\epsilon)$ -approximation algorithm for the CEP in hybrid networks. We developed algorithms with better performance guarantees for special cases of the CEP.
- We extended the approximation algorithm to two capacity expansion problems with additional practical requirements: limiting the growth of certain trunks in one case, and forbidding demand splitting in another.
- For the CEP with survivability requirement, we presented a compact integer

programming formulation that adds just one constraint for every failure scenario.

- We proved that the Survivable CEP is APX-Hard, showing that the existence of a Polynomial Time Approximation Scheme for the problem unlikely.
- Using a decomposition approach, we developed two approximation algorithms with constant factor guarantees. We also developed approximation algorithms with better guarantees for several special cases of the SCEP.

While developing algorithms for the CEP and its variants, we also studied two combinatorial subproblems that are interesting in their own right.

- We studied the Expandable Minimum Knapsack Problem and showed that it is equivalent to the Minimum Knapsack Problem.
- For the Bounded Network Restoration Problem on a two node network, we developed a polynomial time binary search algorithm.

There are many questions that we believe could be the focus of further research. We list some of them here.

- Is there a polynomial time algorithm for the hybrid network CEP with a performance guarantee better than 2?
- Can the analyses and results we presented be extended to more general networks? In particular, can the Decentralized Routing Scheme, which is essentially a cost sharing scheme, be used for approximating the CEP in other network contexts?
- Are there comparable approximation results for the CEP in hybrid networks when more than one facility type is available, i.e., when capacity can be installed in combinations of facilities with capacities C_1, C_2, \dots, C_k ?
- The approximation algorithms we developed for the SCEP assume an empty initial network. Can our results be extended to the case when there are initial capacities on the arcs of the network?

- We showed that unless $P=NP$, the SCEP does not have a PTAS, i.e., there is no $(1+\epsilon)$ -approximation algorithm for all $\epsilon > 0$. But the performance guarantee of the best algorithm we developed is $4 + \epsilon$. Is there an approximation algorithm for the problem? Can we establish stronger inapproximability results?

Capacity Sharing Problem

We proposed a polyhedral model for capturing the variation of demand that is both general and practical. We presented a necessary and sufficient condition for a vector of edge capacities to have feasible flows for every demand in a given demand variation set, and used it to present an integer program for the CSP with infinite constraints. We showed that the problem of separating these constraints is NP-Hard, implying that the CSP is also NP-Hard.

We developed a heuristic procedure for the problem, and evaluated the heuristic computationally on real and random instances. Our experiments suggest that the Capacity Sharing Problem could help telecommunication companies significantly reduce their expenditures for adding capacity. The heuristic we proposed is also effective, producing solutions with cost within 10% of optimum for most instances.

Again, we list a few avenues for further research on the Capacity Sharing Problem.

- Can some of the known valid inequalities for the Capacity Expansion or the Network Loading Problem be extended to the Capacity Sharing Problem? For example, in the cut set formulation for the CSP, we believe valid inequalities similar to residual capacity inequalities for the constraint defining y_e might be used.
- The approximation ratio we established for the integer program with cut constraints was independent of the demand variation set U and therefore was weak. Can we establish stronger bounds for certain specific but practical demand variations?
- For the cutting plane heuristic, we partitioned the commodities by source. But we know that the cutset formulation is valid even if we had grouped two sources

into one subset (union of two stars satisfies Papernov's theorem). Can we develop fast heuristics for the separation problem associated with the two source problem? Would the resulting heuristic be better than the one we proposed in this thesis?

- Our implementation of the cutting plane heuristic spends a substantial amount of time generating the master problem during each iteration, even when we are adding only a single additional constraint. A more efficient implementation, possibly directly using CPLEX libraries, could significantly speed up the heuristic.

Appendix A

APX Hardness Proof for CEP in Hybrid Networks

We present the proof of APX-Hardness, due to Orlin 2005 [44], for the Hub-and-Spoke Network Capacity Expansion Problem (HSP).

Theorem A.1. *The HSP is APX-Hard even if the initial capacities on all the links in the network is zero.*

Proof. We provide a polynomial reduction from the Maximum 3-Dimensional Matching (3DM) problem that is known to be APX-Hard (see Ausiello et al. [8]). In the 3DM problem, we are given three disjoint sets W , X , and Y each containing the same number q of elements, and a set $M \subseteq W \times X \times Y$ of triples with one item each from W , X and Y . We seek to identify the largest subset M' of M such that no two elements of M' agree in any co-ordinate.

Given an instance of 3DM, we create an instance of the HSP as follows. First we describe the hub-and-spoke network. There is a node for every item i in $W \cup X \cup Y$ and for every triple m_j in M . There are two special nodes denoted by s and t and the hub node 0 . Let the j -th triple, m_j in M be (w_j, x_j, y_j) . We create edges $\langle w_j, m_j \rangle$, $\langle x_j, m_j \rangle$, and $\langle y_j, m_j \rangle$, all with facility cost 1. We also create direct edges $\langle m_j, s \rangle$ with cost 2 between every triple m_j in M and the special node s . There are no direct edges out of the special node t . Finally, every node has a radial edge to the hub. For every

item $i \in W \cup X \cup Y$, the edges $\langle i, 0 \rangle$ have cost 4. The radial edges $\langle m_j, 0 \rangle$ also have cost 4 for all triples m_j in M . Finally, the radial edges $\langle s, 0 \rangle$ and $\langle t, 0 \rangle$ out of s and t both have facility cost 0.

The demands between the nodes are as follows. For every triple $m_j = (w_j, x_j, y_j)$ in M , there is a demand of 1 unit each between m_j and w_j , x_j and y_j . For every triple m_j in M , there is a demand of 3 units between m_j and the special node s , and a demand of 1 unit between m_j and the special node t . Finally, there is a demand of 3 units between every item i in $W \cup X \cup Y$ and the node t .

The facility capacity C is 4.

We observe that since there are no direct links out of the node t , in any feasible solution to the HSP, we must route all the demands out of the node t through the hub. Therefore, we will buy one facility each on the radial edges out of the nodes $i \in W \cup X \cup Y$ and one facility each on the radial edges out of the nodes $m_j \in M$ for a total cost of $4q + 4|M|$. This leaves a residual capacity of 1 unit on the radial links out of nodes $i \in W \cup X \cup Y$ and 3 units on the radial links out of nodes $m_j \in M$.

There are four demands out of the node $m_j = (w_j, x_j, y_j)$ to the nodes w_j , x_j , y_j , and S . In any optimal solution to the HSP instance we created, these demands will be satisfied in one of the following two ways:

1. Purchase a facility each on the direct edges $\langle w_j, m_j \rangle$, $\langle x_j, m_j \rangle$, and $\langle y_j, m_j \rangle$ for a total cost of 3 and route these demands directly, and route the demand between m_j and S through the hub using the residual capacities.
2. Route the demands between m_j and the nodes w_j , x_j and y_j through the hub using the residual capacities, and purchase a facility on the direct link $\langle m_j, S \rangle$ for a cost of 2 to route the demand directly.

We refer to the first choice as a Type 1 purchase and the second choice as a Type 2 purchase.

Given an optimal solution to the HSP instance, let M' be the set of triples m_j for which we make a Type 2 purchase. Then the cost of this solution is $(4q + 4|M|) +$

$(2|M'| + 3(|M| - |M'|))$. We note that the set M' is a feasible solution to the 3DM problem.

On the other hand, given an optimal solution M' to the 3DM problem, we can make Type 2 purchases for all the triples in M' , and Type 1 purchases for all other triples to obtain a feasible solution to the HSP with a cost of $(4q + 4|M|) + (2|M'| + 3(|M| - |M'|))$.

We conclude that the 3DM problem has an optimal solution with k triples if and only if the optimal cost of the HSP is $(4q + 4|M|) + (3|M| - k)$. This completes our reduction from the 3DM problem to the HSP without initial capacities. \square

Bibliography

- [1] Cisco - Voice design and implementation guide. Technical report, Cisco Systems, Inc., San Jose, CA, 2005.
- [2] D. Alevras, M. Grötschel, and R. Wessäly. *Capacity and Survivability Models for Telecommunication Networks*. ZIB Preprint SC 97-24, ZIB, Berlin, 1997.
- [3] D. Alevras, M. Grötschel, and R. Wessäly. Cost-efficient network synthesis from leased lines. *Annals of Operations Research*, 76:1–20, 1998.
- [4] G. R. Ash. Dynamic network evolution, with examples from AT&Ts evolving dynamic network. *Communications Magazine, IEEE*, 33(7):26–39, 1995.
- [5] G. R. Ash. *Dynamic Routing in Telecommunications Networks*. McGraw-Hill, 1998.
- [6] G. R. Ash and E. Oberer. Dynamic routing in the AT&T network improved service quality at lower cost. *GlobeCom'89, IEEE*, 1:303–308, 1989.
- [7] A. Atamtürk and M. Zhang. Two-stage robust network flow and design under demand uncertainty. *Operations Research*, 55(4), 2007.
- [8] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and Approximation: Combinatorial Optimization Problems and their Approximability Properties*. Springer Verlag, 1999.
- [9] A. Balakrishnan, T. L. Magnanti, A. Shulman, and R. T. Wong. Models for planning capacity expansion in local access telecommunication networks. *Annals of Operations Research*, 33:239–284, 1991.

- [10] A. Balakrishnan, T. L. Magnanti, J. S. Sokol, and Y. Wang. Telecommunication link restoration planning with multiple facility types. *Annals of Operations Research*, 107(1–4):127–154, 2001.
- [11] A. Balakrishnan, T. L. Magnanti, J. S. Sokol, and Y. Wang. Spare-capacity assignment for line restoration using a single-facility type. *Operations Research*, 50(4):617–635, July-August 2002.
- [12] 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, Jan.-Feb. 1995.
- [13] A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski. Adjustable robust solutions of uncertain linear programs. *Mathematical Programming*, 99(2):351–376, 2004.
- [14] D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Mathematical Programming, Series B*, 98:49–71, 2003.
- [15] D. Bienstock, S. Chopra, O. Gunluk, and C. Tsai. Minimum cost capacity installation for multicommodity network flows. *Mathematical Programming*, 81:177–199, 1998.
- [16] D. Bienstock and O. Gunluk. Capacitated network design-polyhedral structure and computation. *INFORMS Journal on Computing*, 8(3):243–259, 1996.
- [17] D. Bienstock and G. Muratore. Strong inequalities for capacitated survivable network design problems. *Mathematical Programming, Ser. A*, 89:127–147, 2000.
- [18] S. Chang and B. Gavish. Lower bounding procedures for multiperiod telecommunications network expansion problems. *Operations Research*, 43(1):43–57, Jan.-Feb. 1995.
- [19] M. P. Clark. *Networks and Telecommunications*. John Wiley and Sons, Ltd., second edition, 1997.

- [20] G. B. Dantzig, L. R. Ford, and D. R. Fulkerson. *A primal-dual algorithm for linear programs*, chapter Linear Inequalities and Related Systems, pages 171–181. Annals of Mathematical Studies 38. Princeton University Press, Princeton, New Jersey, 1956.
- [21] A. Z. Dodd. *The Essential Guide to Telecommunications*. Prentice Hall PTR, 4th edition, 2005.
- [22] R. Epstein. *Linear Programming and Capacitated Network Loading*. PhD thesis, Operations Research Center, Massachusetts Institute of Technology, Cambridge, MA, 1998.
- [23] B. Gavish. Topological design of telecommunication networks - local access design methods. *Annals of Operations Research*, 33:17–71, 1991.
- [24] G. V. Gens and E. V. Levner. Computational complexity of approximation algorithms for combinatorial problems. In *Mathematical Foundations of Computer Science*, volume 74 of *Lecture Notes in Computer Science*, pages 292–300. 1979.
- [25] J. H. Green. *The Irwin Handbook of Telecommunications*. McGraw-Hill, fifth edition, 2006.
- [26] M Grötschel, C. Monma, and M. Stoer. *Design of Survivable Networks*, volume 7, chapter Handbooks of Operations Research and Management Science: Network Models., pages 617–672. North-Holland, Amsterdam, The Netherlands, 1995.
- [27] A. Gupta, A. Kumar, and T. Roughgarden. Simpler and better approximation algorithms for network design. In *STOC '03: Proceedings of the Thirty-fifth Annual ACM Symposium on Theory of Computing*, pages 365–372, New York, NY, USA, 2003. ACM Press.
- [28] R. Hassin, R. Ravi, and F. S. Salman. Approximation Algorithms for a Capacitated Network Design Problem. *Algorithmica*, 38(3):417–431, Dec. 2003.

- [29] O. H. Ibarra and C. E. Kim. Fast approximation algorithms for the knapsack and sum of subset problem. *Journal of the ACM*, 22:463–468, 2002.
- [30] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems*. Springer, 2004.
- [31] J. L. Kennington, E. V. Olinick, and G. Spiride. Basic mathematical programming models for capacity allocation in mesh-based survivable networks. *Omega*, 2006.
- [32] B. Korte and J. Vygen. *Combinatorial Optimization, Theory and Algorithms*. Springer-Verlag, Berlin, 2002.
- [33] P. Kouvelis and G. Yu. *Robust Discrete Optimization and Its Applications*. Kluwer Academic Publishers, 1997.
- [34] A. Lissner, R. Sarkissian, and J. Vial. Optimal joint syntheses of base and spare telecommunication networks. *International Symposium on Mathematical Programming*, 1997.
- [35] H. Luss. Operations research and capacity expansion problems: A survey. *Operations Research*, 30(5):907–947, Sep.-Oct. 1982.
- [36] T. L. Magnanti, P. Mirchandani, and R. Vachani. The convex hull of two core capacitated network design problems. *Mathematical Programming*, 60(2):233–250, June 1993.
- [37] T. L. Magnanti, P. Mirchandani, and R. Vachani. Modeling and solving the two-facility capacitated network loading problem. *Operations Research*, 43(1):142–157, Jan.-Feb. 1995.
- [38] T. L. Magnanti and Y. Wang. Polyhedral properties of the network restoration problem – With the convex hull of a special case. Working paper OR 323-97, Operations Research Center, Massachusetts Institute of Technology, Cambridge, MA, 1997.

- [39] Y. Mansour and D. Peleg. An Approximation Algorithm for Minimum-Cost Network Design. Technical Report CS94-22, The Weizman Institute of Science, Rehovot, Israel, 1994.
- [40] S. Martello and P. Toth. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley and Sons, Ltd., 1990.
- [41] D. Medhi. A unified approach to network survivability for teletraffic networks: Models, algorithms and analysis. *IEEE Transactions on Communications*, 43(234):534–548, Feb./Mar./Apr. 1994.
- [42] S. Mudchanatongsuk, F. Ordonez, and J. Liu. Robust solutions for network design under transportation cost and demand uncertainty. *Working Paper*, 2007.
- [43] K. Onaga. A multi-commodity flow theorem [english translation]. *Electronics and Communications in Japan*, 53-A(7):16–22, 1970.
- [44] J. B. Orlin. Personal communication, 2005.
- [45] B. A. Papernov. Realizuemost' mnogoproductovykh potokov [Russian: Feasibility of multicommodity flows]. *Issledovaniya po Diskretnoi Optimizatsii [Russian: Studies in Discrete Optimization]*, pages 230–261, 1976.
- [46] H. Sakauchi, Y. Nishimura, and S. Hasegawa. A self-healing network with an economical spare-channel assignment. *Globecom'90*, pages 438–443, 1990.
- [47] F. S. Salman, J. Cheriyan, R. Ravi, and S. Subramanian. Approximating the Single-Sink Link-Installation Problem in Network Design. *SIAM Journal on Optimization*, 11(3):595–610, 2001.
- [48] I. Saniee. An efficient algorithm for the multiperiod capacity expansion of one location in telecommunications. *Operations Research*, 43(1):187–190, Jan.-Feb. 1995.
- [49] A. Schrijver. *Combinatorial Optimization - Polyhedra and Efficiency*. Springer, 2003.

- [50] A. Shulman and R. Vachani. A decomposition algorithm for capacity expansion of local access networks. *IEEE Transactions on Communication*, 41(7):1063–1073, July 1993.
- [51] M. Stoer and G. Dahl. A polyhedral approach to multicommodity survivable network design. *Numerische Mathematik*, 68(1):149–167, 1994.
- [52] J. Veerasamy, S. Venkatesan, and J.C. Shah. Spare capacity assignment in telecom networks using path restoration. *Mascots'95*.
- [53] Y. Wang. *Modeling and Solving Single and Multiple Facility Network Restoration Problems*. PhD thesis, Operations Research Center, Massachusetts Institute of Technology, Cambridge, MA, 1998.