DESIGNING THE LIVER ALLOCATION HIERARCHY: INCORPORATING EQUITY AND UNCERTAINTY

by

Mehmet C. Demirci

B.S., Middle East Technical University, 2003

M.S., University of Pittsburgh, 2005

Submitted to the Graduate Faculty of

the Swanson School of Engineering in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2008

UNIVERSITY OF PITTSBURGH

SWANSON SCHOOL OF ENGINEERING

This dissertation was presented

by

Mehmet C. Demirci

It was defended on

May 2, 2008

and approved by

Andrew J. Schaefer, Associate Professor, Department of Industrial Engineering

Brady Hunsaker, Software Engineer, Google

Jayant Rajgopal, Associate Professor, Department of Industrial Engineering

Prakash Mirchandani, Professor, Katz Graduate School of Business

Mark S. Roberts, Professor, School of Medicine

Dissertation Director: Andrew J. Schaefer, Associate Professor, Department of Industrial Engineering

DESIGNING THE LIVER ALLOCATION HIERARCHY: INCORPORATING EQUITY AND UNCERTAINTY

Mehmet C. Demirci, PhD

University of Pittsburgh, 2008

Liver transplantation is the only available therapy for any acute or chronic condition resulting in irreversible liver dysfunction. The liver allocation system in the U.S. is administered by the United Network for Organ Sharing (UNOS), a scientific and educational nonprofit organization. The main components of the organ procurement and transplant network are Organ Procurement Organizations (OPOs), which are collections of transplant centers responsible for maintaining local waiting lists, harvesting donated organs and carrying out transplants. Currently in the U.S., OPOs are grouped into 11 regions to facilitate organ allocation, and a three-tier mechanism is utilized that aims to reduce organ preservation time and transport distance to maintain organ quality, while giving sicker patients higher priority. Livers are scarce and perishable resources that rapidly lose viability, which makes their transport distance a crucial factor in transplant outcomes. When a liver becomes available, it is matched with patients on the waiting list according to a complex mechanism that gives priority to patients within the harvesting OPO and region. Transplants at the regional level accounted for more than 50% of all transplants since 2000.

This dissertation focuses on the design of regions for liver allocation hierarchy, and includes optimization models that incorporate geographic equity as well as uncertainty throughout the analysis. We employ multi-objective optimization algorithms that involve solving parametric integer programs to balance two possibly conflicting objectives in the system: maximizing efficiency, as measured by the number of viability adjusted transplants, and maximizing geographic equity, as measured by the minimum rate of organ flow into

individual OPOs from outside of their own local area. Our results show that efficiency improvements of up to 6% or equity gains of about 70% can be achieved when compared to the current performance of the system by redesigning the regional configuration for the national liver allocation hierarchy.

We also introduce a stochastic programming framework to capture the uncertainty of the system by considering scenarios that correspond to different snapshots of the national waiting list and maximize the expected benefit from liver transplants under this stochastic view of the system. We explore many algorithmic and computational strategies including sampling methods, column generation strategies, branching and integer-solution generation procedures, to aid the solution process of the resulting large-scale integer programs. We also explore an OPO-based extension to our two-stage stochastic programming framework that lends itself to more extensive computational testing. The regional configurations obtained using these models are estimated to increase expected life-time gained per transplant operation by up to 7% when compared to the current system.

This dissertation also focuses on the general question of designing efficient algorithms that combine column and cut generation to solve large-scale two-stage stochastic linear programs. We introduce a flexible method to combine column generation and the L-shaped method for two-stage stochastic linear programming. We explore the performance of various algorithm designs that employ stabilization subroutines for strengthening both column and cut generation to effectively avoid degeneracy. We study two-stage stochastic versions of the cutting stock and multi-commodity network flow problems to analyze the performances of algorithms in this context.

Keywords: Organ allocation, large-scale medical optimization, column generation, branch-and-price, multi-objective decision making, two-stage stochastic programming.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

This study is dedicated to my wonderful family: Hülya and Nuri Demirci, and Ayşegül Demirci Çoban.

I would like to express my gratitude to my advisor, Dr. Andrew Schaefer, for his guidance and for supporting me through my doctoral studies. I also would like to thank the rest of my dissertation committee, Drs. Brady Hunsaker, Jayant Rajgopal, Prakash Mirchandani and Mark Roberts, for their invaluable suggestions and insights.

I am grateful to my friends who made my stay in Pittsburgh a very enjoyable experience. I would especially like to thank Marlene DeAngelo, Görkem Saka, Halil Bayrak, Alp Şekerci, Nuri Mehmet Gökhan, Tuba Pınar Yıldırım, Mustafa Baz, Erkut Sönmez, Rob Koppenhaver, Chris Roth, Anıl Yılmaz, Zeynep Erkin, Gözde İçten, Sakine Batun, Başak Yavçan, Özlem Arısoy, Tuğba Özkasap, Burhaneddin Sandıkçı, Murat Kurt, Osman Özaltın and Işıl Öndeş.

I wish to thank Drs. Jay Rosenberger and Edwin Romeijn, with whom I had the chance to collaborate, for their input and valuable discussions. I also acknowledge the financial support provided by National Science Foundation Grant DMI-0355433.

Many thanks go to Richard Brown, Minerva Pilachowski, Nora Siewiorek and Jim Segneff for providing technical support throughout my graduate studies.

Most importantly, I am forever indebted to my parents Hülya and Nuri Demirci, my sister Ayşegül Demirci Çoban and my nephew Efe Çoban. None of this would have been possible without their endless love, inspiration, encouragement and unconditional support. I am very lucky to have such a wonderful family!

1.0 INTRODUCTION

Health care continues to be a rising issue in the United States today. The U.S. health care spending has historically shown significant growth over the years, amounting to $2.1 trillion in 2006, which is equivalent to 16 percent of the nation's Gross Domestic Product (GDP) or $7,026 per person [30]. Currently, both on a per-capita basis and as a fraction of the GDP, the U.S. spends more on health care than any other country in the world [180]. Furthermore, the health care services share of the GDP is projected to reach 19.5 percent by the end of 2017, with a nominal expenditure of over $4.3 trillion [31]. Despite the increasing burden of health care spending on the national economy, the World Health Organization (WHO) ranked the the U.S. 37^{th} in health care system performance and 72^{nd} in overall level of health among the 191 member countries in the World Health Report 2000 [181].

Concerns about the performance of the U.S. health care system and the increasing proportion of health care expenses to the nation's GDP have spurred a growing interest in the field of medical decision making over the last few decades. Numerous researchers have utilized optimization techniques to address the questions regarding how to design, schedule or manage health care systems. The applications of operations research in health care range from policy or systems design studies, like personnel scheduling [90, 120, 178], ambulance location [26, 28], emergency room or operating room scheduling [36, 109], organ allocation policy design [96, 97, 157], to the treatment of individual patients, such as cancer treatment [103, 107, 112, 135, 182, 188], and optimal timing of organ transplants [2, 4, 5, 6, 45, 80, 81, 150].

Donated human organs for transplantations are scarce and highly perishable resources that have to be allocated efficiently in order to maximize potential outcomes and minimize waste. Despite the fact that the number of organ transplants (kidney, liver, heart, lung,

pancreas or intestine) have been steadily increasing over the last 20 years, the number of donations and transplants have not kept pace with the increase in the size of the waiting list for organs [169]. For instance, in the U.S., 28,354 transplants were made within the year 2007 while 7,028 patients died while waiting for an organ, and 105,850 patients were waiting for a transplant as of April 2008 [169].

In this dissertation, we focus on End-stage Liver Disease (ESLD) and the liver allocation system in the U.S. End-stage liver disease is any acute or chronic condition resulting in irreversible liver dysfunction, and includes diseases such as chronic liver disease, primary biliary cirrhosis and hepatitis. Unlike other organs such as kidney, which has dialysis as an alternative therapy, the only available therapy for ESLD is liver transplantation. ESLD has consistently been a leading cause of death in the U.S. according to the National Vital Statistics Reports published by the National Center for Health Statistics (NCHS). As can be seen from Table 1.1, which shows the leading causes of death in the U.S. for the year 2004 [126], ESLD was the twelfth leading cause of death in the U.S. in 2004, claiming nearly 30,000 lives.

Table 1.2 shows the liver transplant waiting list related data between the years 2000 and 2007 [169]. As can be seen from this table, the number of donations and transplants have not kept pace with the increase in the size of the ESLD patient population registered on the national waiting list. In addition to around 2,000 patient deaths while waiting for a liver, it is observed that almost 10% of the harvested livers were wasted. The trend in the national waiting list between years 1996 and 2007 can be seen in Figure 1.1. These numbers suggest the need for improvements in liver allocation system performance.

After receiving a liver transplant, ESLD patients can expect 1-year and 5-year survival rates of around 90% and 78%, respectively. Table 1.3 shows the post-transplant patient survival rates in the U.S. between 1997-2004. [169]. One critical factor that directly affects transplant outcomes is the time the donated organ spends outside of a human body. *Cold ischemia time (CIT)* is the time interval that begins when an organ is cooled with a cold perfusion solution after organ procurement surgery and ends when the organ is implanted [171]. Livers rapidly lose quality with time spent outside of the human body. Longer distances between recipients and donated organs generally mean higher CITs, and hence,

Table 1.1: Leading causes of death in the U.S. in 2004 according to the National Vital Statistics Report [126].

Rank	Cause of death	Number	Percentage
-	All causes	2,397,615	100.0
1	Diseases of heart	652,486	27.2
2	Malignant neoplasms	553,888	23.1
3	Cerebrovascular diseases	150,074	6.3
4	Chronic lower respiratory diseases	121,987	5.1
5	Accidents (unintentional injuries)	112,012	4.7
6	Diabetes mellitus	73,138	3.1
7	Alzheimers disease	65,965	2.8
8	Influenza and pneumonia	59,664	2.5
9	Nephritis, nephrotic syndrome and nephrosis	42,480	1.8
10	Septicemia	33,373	1.4
11	Intentional self-harm (suicide)	32,439	1.4
12	*Chronic liver disease and cirrhosis*	*27,013*	*1.1*
13	Essential hypertension and hypertensive renal disease	23,076	1.0
14	Parkinsons disease	17,989	0.8
15	Assault (homicide)	17,357	0.7
-	All other causes	414,674	17.3

Table 1.2: U.S. liver data between 2000-2007 [169].

	2000	2001	2002	2003	2004	2005	2006	2007
Patients waiting	16,440	18,269	17,072	17,310	17,411	17,673	17,523	17,167
Patients added	10,751	10,741	9,329	10,046	10,639	10,986	11,036	11,081
Deaths	1,823	2,082	1,935	1,889	1,922	1,925	1,782	1,602
Donations								
All donors	5,399	5,628	5,657	6,004	6,642	7,015	7,305	7,204
Cadaveric	4,997	5,106	5,294	5,682	6,319	6,692	7,017	6,939
Living	402	522	363	322	323	323	288	265
Transplants								
All donors	4,997	5,195	5,332	5,673	6,169	6,442	6,650	6,492
Cadaveric	4,595	4,673	4,969	5,351	5,846	6,119	6,362	6,227
Living	402	522	363	322	323	323	288	265
Wasted organs[a]	402	433	325	331	473	573	655	712

[a]A recovered liver is assumed to be wasted if it is donated by a deceased donor but is not used for transplantation

decay in transplant quality. While the maximum acceptable CIT for livers is 18-24 hours [39], most livers have a CIT less than 10 hours [171]. Hence, the liver allocation system in the U.S. today aims to reduce organ preservation time and transport distance to maintain organ quality, while giving sicker patients higher priority.

1.1 LIVER ALLOCATION IN THE U.S.

The liver allocation system in the U.S. is administered by the United Network for Organ Sharing (UNOS) [169], a scientific and educational nonprofit organization. The organ procurement and sharing network is composed of Organ Procurement Organizations (OPOs), which are responsible for the identification and care of organ donors, and also organ retrieval, preservation, transportation and transplantation. OPO staff work with donor families, and educate medical staff and general public about organ donation [165].

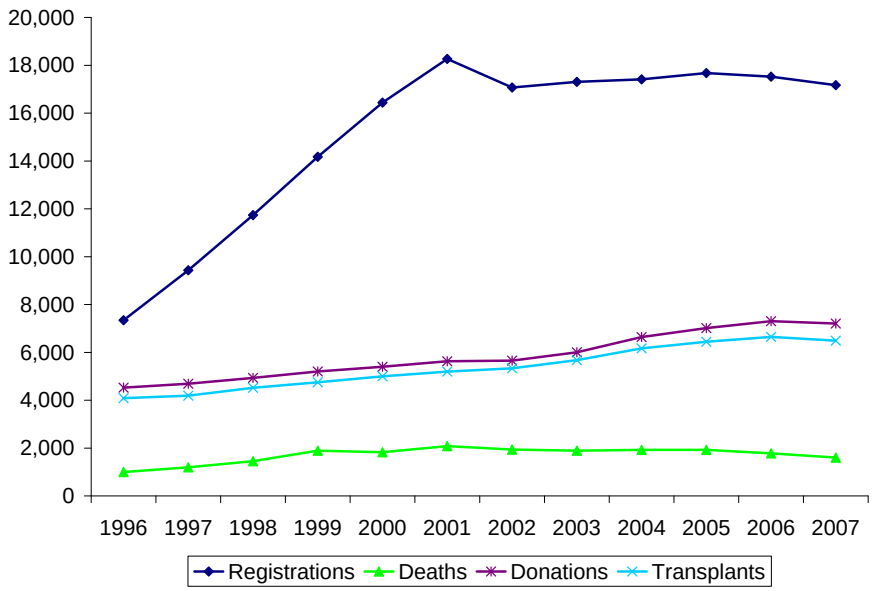


Figure 1.1: Liver transplant waiting list trends in recent years [169].

UNOS assesses the severity of ESLD in adults using the *Model for End-Stage Liver Disease (MELD)*, a scoring system for chronic liver disease [88, 113, 179]. The MELD score is a function of total bilirubin, creatinine and prothrombin times. It was first introduced by Malinchoc et al. [113] to evaluate the prognosis for liver cirrhosis patients. Wiesner et al. [179] present the formula for MELD as follows:

$$\text{MELD} = 9.57 \times \ln(\text{creatinine mg/DL}) + 3.78 \times \ln(\text{bilirubin mg/DL}) + 11.2 \times \ln(\text{INR})$$

$$+6.43 \times (\text{constant for liver disease etiology}),$$

where INR, i.e., the international normalized ratio, is obtained by dividing the patient prothrombin time by a normal prothrombin time value, and the constant for liver disease etiology is 1 if the liver disease is alcohol or cholestatic related, and is 0 otherwise. UNOS uses a modified version of the MELD score for the national liver allocation system [169]. As used by UNOS, the MELD score of a patient is an integer between 6 and 40, and higher scores

Table 1.3: Kaplan-Meier patient survival rates for liver transplants performed: 1997 - 2004 [169].

Donor Type	Survival rate (%)		
	1-year	3-year	5-year
Cadaveric	86.3	78.0	72.1
Living	90.1	82.7	77.8

indicate greater sickness. *Pediatric End-Stage Liver Disease (PELD)* is a similar model to assess the severity of liver disease in pediatric patients. UNOS employs different procedures for adult and pediatric patients. In this dissertation we focus on the liver allocation system for adults and do not consider patients under the age of 18.

Acutely ill patients with expected life times of less than one week without a transplant are called *Status 1* patients. After a policy change for Status 1 listing in August 2005, transplant candidates must have fulminant hepatic failure, primary nonfunction of a transplanted liver, hepatic artery thrombosis or acute decompensated Wilson's disease to be listed as *Status 1A* [169]. *Status 1B* is exclusively used for pediatric patients with acute decompensation of chronic liver disease. All other patients who are not assigned to a Status 1 level are called MELD patients. Status 1 patients only make up less than 0.1% of the ESLD patient population [169].

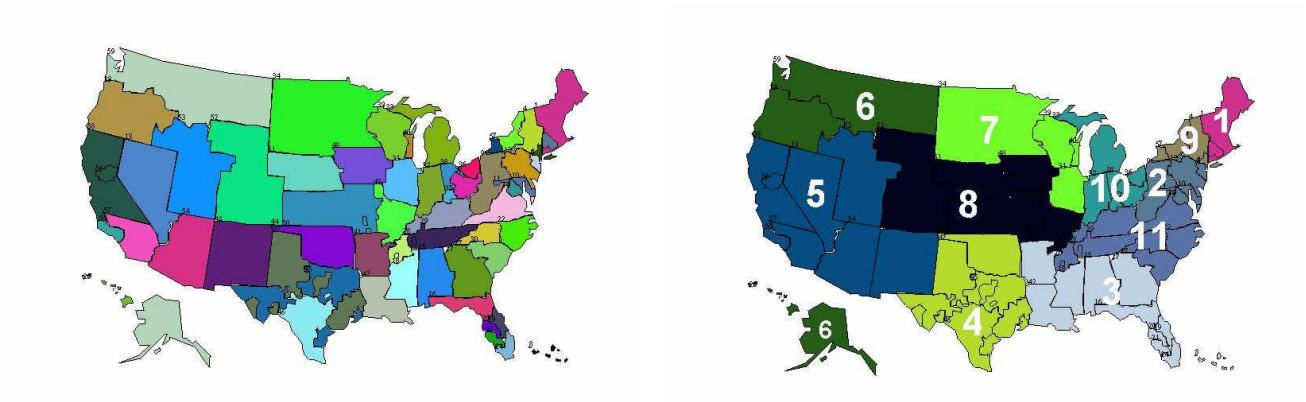
The UNOS organ allocation policies have been a subject of debate over the past years. In the center of concerns regarding the organ allocation system lies the efficiency of the allocation mechanism. There is trade off between encouraging local usage of organs and allowing organ sharing across different regions of the country. Since transplantable organs are highly perishable resources, local usage of organs are preferable due to the fact that the expected transplant outcomes would be better. However, organ sharing across regions increases the likelihood of finding better patient-organ matches.

Consequently, two different allocation approaches have been advocated in the past years. The first approach emphasizes the importance of medical urgency and advocates a single national waiting list where patients are ranked on their medical urgency, and no location criteria is used while ranking the patients on the list. Since the CIT of a liver can go up to 18-24 hours [39], national matching, in fact, is feasible for most cases. However, since the livers lose quality with the time spent outside of a human body, transplant outcomes will be worse if the organ has to be transported over long distances for transplantation. Hence, a second approach focuses on the benefits of using organs locally due to lower CIT and advocates local usage of organs. A possible shortcoming of enforcing local usage of organs is the relatively low population pool for organ matching which might decrease the chances of identifying good patient-organ matches. In an attempt to balance these concerns, UNOS designed an allocation mechanism that is driven by both medical urgency and patient location. Organ transplant regions were introduced to enable organ sharing across larger areas while maintaining organ quality.

When a liver becomes available for transplantation, patients on the national waiting list are ranked according to UNOS's allocation mechanism. The ranking mechanism takes into account factors such as the size of the organ and patient, medical urgency, blood type, tissue type, waiting time on the list and patient location. After the list of potential recipients is obtained, the transplant surgeons responsible for the patients on the list are contacted. The transplant team is given one hour to decide whether to accept a liver offer or not [170].

The two most important factors affecting UNOS's liver allocation hierarchy are the severity of illness and the geographic locations of potential recipients and the available organ. UNOS liver allocation policies aim to reduce organ preservation time and transport distance to maintain organ quality, while giving sicker patients higher priority. The liver allocation system is based on a three-tier mechanism in which the patients within the service area of the donor OPO have the top priority, followed by patients listed in the region of the donor OPO, and finally, all patients nationwide.

The liver allocation system has been revised several times in the past decade, and UNOS has also added some differentiations among both Status 1 and MELD patients that modified the aforementioned three-tier framework [72, 169]. However, the underlying priorities in the



(a) (b)

Figure 1.2: (a) Current OPO service areas across the nation. (b) Current regional configuration in the U.S.

main three-tier mechanism with respect to the geographic locations of donors and recipients has remained intact. We refer to UNOS Policies and Bylaws [168] for further details on the liver allocation mechanism. The steps of the allocation mechanism for adult ESLD patients can be summarized as follows:

Step 1. Status 1 patients in the designated service area of the donor OPO.

Step 2. Status 1 patients in the region of the donor OPO.

Step 3. MELD patients in the designated service area of the donor OPO.

Step 4. MELD patients in the region of the donor OPO.

Step 5. All remaining Status 1 patients nationwide.

Step 6. All remaining MELD patients nationwide.

Figure 1.3, which is from Kong [96], shows a schematic representation of the allocation hierarchy.

There are currently 58 OPOs in the U.S., which are grouped into 11 regions. Maps of current OPOs and regions can be found in Figure 1.2. The regional structure was developed to facilitate organ allocation and provide individuals with the opportunity to identify concerns regarding organ procurement, allocation and transplantation that are unique to their particular geographic area [169]. In the three-tier system that UNOS manages, more than 50% of the total transplants occur in the intra-regional level [169]. The design of the

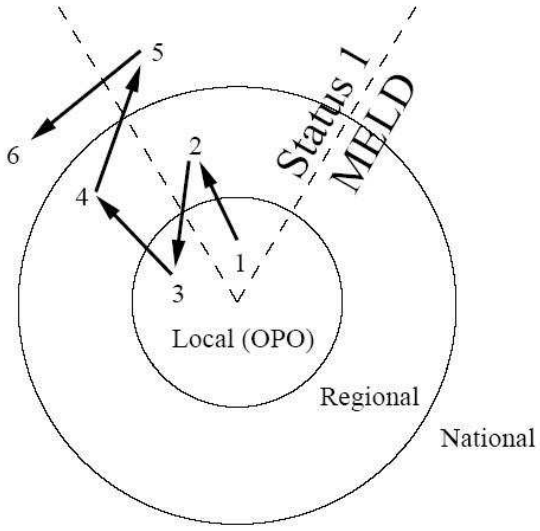


Figure 1.3: UNOS liver allocation hierarchy [96].

regions is a crucial factor in the efficiency of these intra-regional transplants. Ideally, a region should be large enough to include sufficiently populated areas to increase the chances of donor-recipient matches, but also compact enough so that organ transport times within the region, which in turn directly affect organ and transplant quality, would be acceptable. The current regions were put in place by the UNOS Board of Directors but no details were released on quantitative analysis used throughout the process.

1.2 PROBLEM STATEMENT AND PROPOSED RESEARCH DESCRIPTION

In this dissertation, we focus on the design of regions for the liver allocation hierarchy. We analyze the liver allocation system under different objectives using multi-objective integer programming and two-stage stochastic mixed-integer programming techniques. We seek to maximize organ utilization and transplant benefits, and minimize organ wastage through redesigning the regional configuration of OPOs across the U.S.

A common aspect of these models is that they all involve branch-and-price based solution algorithms. We introduce set-partitioning based mathematical models to divide the nation into liver transplant regions where binary variables represent different region designs, and the objective is to find the optimal partitioning of OPOs into a distinct set of regions. We utilize column generation to generate promising regions as needed throughout the solution procedure. For parameter estimation and solution validation, we modify the ESLD and liver allocation system simulation model of Shechter et al. [155].

Throughout this dissertation, we focus our attention to MELD patients on the waiting list and ignore Status 1 patients. The reason behind this is that Status 1 patients only make up less than 0.1% of the ESLD patient population [169], and incorporating them into our modeling framework complicates the model structure. When Status 1 patients are not taken into account, Steps 1 and 2 in Figure 1.3 can be eliminated, and the local matching phase only consists of Step 3. Moreover, since the intra-OPO transplants are independent of region design, there is no need to consider this phase in the model.

Allocation of organs at the national level, on the other hand, poses some additional modeling problems. The flow of organs among regions at the national level depends on the regional configuration of the country. Since the proposed set-partitioning based models cannot explicitly incorporate the interdependency of regions, and the percentage of transplants that occur at the national level is only around 5% [169], we do not fully model Steps 5 and 6 in Figure 1.3, but rather capture the effects of national level allocation implicitly in our model parameters.

Hence, the main focus of our analysis is on the regional matching phase for MELD patients in UNOS's liver allocation mechanism, i.e., Step 4 in Figure 1.3. Previous studies on region design [96, 97, 157] also relied on similar simplifications, and focused on the regional organ allocation phase for MELD patients. The framework we adopt only focuses on the design of regions for liver transplantation. We aim to maximize the national benefit by redesigning the regions without interfering with any UNOS policy regarding how patients should be prioritized.

Various performance measures can be monitored for the liver allocation system, the most prominently studied one being the *efficiency* as measured by the number of transplants. An

alternative view of the efficiency of the system focuses on the expected benefit gained per patient following a transplant, in terms of the life-time gained. We also study the minimum *geographic equity* as measured by the patients' access to organs from outside their own local area.

1.3 CONTRIBUTION

In this dissertation, we extend previous studies on liver transplant region design [96, 97, 157] in a number of ways. First, we construct a multi-objective optimization model using refined estimates of geographic equity that takes into account the probability of organ flow between OPO pairs, and solve the model using exact methods, in contrast to the approximate methods used in previous studies [96, 157]. Furthermore, according to our validation tests using the liver allocation system simulation model of Shechter et al. [155], our proposed regional configurations outperform the alternative configurations provided by previous studies [97, 157] in almost all cases. We also design two-stage stochastic programming models that focus on the uncertainty of the system, in contrast to the steady-state analysis of previous studies, and adopt the objective of maximizing the expected outcome of transplants, as measured by the life-time gained by the recipients, which has never been studied in a region-design context. Motivated by the structure of our stochastic programming models, we also investigate the general question of how to design effective algorithms that combine column generation with the L-shaped method [22, 174] to solve two-stage stochastic linear programs that involve too many variables to handle explicitly. We compare our work to existing literature in more detail in Sections 2.3 and 2.4.

The rest of this dissertation is organized as follows: Chapter 2 presents a review of the relevant literature on operations research applications in health care optimization. We particularly focus on the modeling and solution methodologies utilized in this dissertation and their applications, primarily in the area of medical decision making. Chapter 3, introduces parametric integer programming models for balancing efficiency and equity while designing regions for the liver allocation system. The chapter discusses two exact algorithms that use

these parametric models to approximate the efficient frontier of the system under two possibly conflicting objectives: maximizing efficiency and maximizing equity. Chapter 4 relaxes the assumption of a steady-state system that all previous region design studies utilized in the modeling phase, and adopts a stochastic view of the liver allocation mechanism in which the objective is to maximize the outcome of liver transplants as measured by the expected life-time gained. The chapter utilizes a column generation approach through the use of a pricing problem that can handle the ranking of individual patients on the waiting list under different scenarios. Chapter 5 modifies the stochastic programming framework of Chapter 4 by adopting an OPO-based approach in which patients in every OPO are replaced by aggregate patients. This modification helps scale the problem down and allows for more intensive computational testing using larger scenario samples. Chapter 6 generalizes the solution techniques used in Chapters 4 and 5, and introduces a decomposition approach that combines column generation and the L-shaped method to solve two-stage stochastic linear programs. We discuss convergence and algorithmic variants, and utilize two-stage stochastic versions of the cutting-stock and multi-commodity flow problems for computational experiments. Finally, Chapter 7 summarizes our conclusions and discusses potential future research directions.

2.0 LITERATURE REVIEW

In this chapter, we review the literature related to the problems and methodologies discussed in this dissertation. In Section 2.1, we discuss the technique of integer programming and review some applications, particularly in heath care. We also briefly discuss branch and price and column generation techniques. Section 2.2 reviews multi-objective programming techniques and applications with a focus on medical decision making applications. In Section 2.3 we present a survey on stochastic programming and its applications in health care optimization. Section 2.4 presents a survey regarding studies related to organ allocation and transplantation systems. We focus on operations research literature on this subject, in particular that describing liver transplant region design.

2.1 INTEGER PROGRAMMING (IP)

Integer programming has been a popular modeling framework for a wide variety of applications. Although integer programs are much harder to solve than linear programs, they provide decision makers with modeling opportunities that can closely match modeling requirements for real-world systems. Integer programs have been successfully applied to many planning, scheduling, routing, etc., settings in various areas. We refer to Nemhauser and Wolsey [128] for more information on modeling and solution techniques in integer programming.

2.1.1 IP Applications in Health Care

In this section, we present a representative set of publications that summarize integer programming applications in health care optimization. For comprehensive surveys of operations research applications in health care, we refer to Brandeau et al. [27], Griffin et al. [73] and Pierskalla and Brailer [133], which also include numerous references on integer programming applications.

The location of health care facilities has drawn considerable attention in the operations research literature. A recent survey of applications in health care facility location is provided by Daskin and Dean [44], which lists numerous applications of classic location models, such p-median and set covering models, in health care facility location [43]. Location applications in health care include determining hospital locations [117, 156], ambulance locations [1, 26, 28, 57, 116], blood banks [85, 136], and emergency services [18, 56, 68, 125, 134, 152].

Integer programming has also been used for applications in cancer treatment, like radiosurgery treatment planning [103]. Several recent studies have focused on brachytherapy treatment planning and IMRT. Brachytherapy is a form of radiotherapy where a radioactive source is placed inside or next to the area requiring treatment, and is commonly used for treating prostate cancer, and cancers of the head and neck. Integer programming applications in brachytherapy treatment include [104, 105, 106, 107, 183]. Intensity Modulated Radiotherapy Treatment (IMRT) is the medical use of radiation as part of cancer treatment to control malignant cells in a cancer patient. IMRT design problems focus on how radiation beams will travel through a patient so that a tumoricidal radiation dose will be effectively delivered to the cancerous region with minimal damage to healthy organs and tissues. Recent studies that utilized IP for IMRT design include [100, 101, 135, 145].

Integer programming models have also been utilized in areas like personnel scheduling [90, 120, 178], organ allocation policy design [96, 97, 157], and vaccine selection and design [86, 87, 153].

2.1.2 Branch and Price

As mentioned in Chapter 1, column generation and branch-and-price techniques are the underlying solution methods in every chapter of this dissertation. In this section, we briefly introduce these methods and discuss a few applications.

Column generation methodology is commonly used to solve mathematical problems that involve a large number of variables. Instead of considering all of the columns of such a problem explicitly, column generation techniques employ a master problem that contains a subset of columns and a pricing subproblem that generates promising new columns. The algorithm follows a loop in which the master problem is solved, and the dual solution is passed on to the pricing problem, which tries to find the column with the most favorable reduced cost. If favorable columns exist, some subset of them are inserted into the master problem; if not, the algorithm stops. Column generation was first introduced by Ford and Fulkerson [64] to enable the implicit handling of variables in a multi-commodity flow problem. Dantzig and Wolfe [42] helped establish column generation as a powerful technique for large-scale mathematical programming by utilizing this method in the algorithm known as Dantzig-Wolfe decomposition.

Dantzig-Wolfe decomposition [42] is a technique that relies on column generation. In this technique a set of "easy" constraints are replaced by variables. Each variable corresponds to an extreme point or ray of the polyhedron defined by the "easy" constraints. Column generation adds variables that have favorable reduced costs as needed. The master problem (MP) contains a subset of all possible variables, and the dual solution of the MP is passed on to the pricing subproblem. If there exists a variable that improves the objective value, it is added to the MP; if not, the algorithm terminates.

Column generation has been successfully applied to an enormous number of deterministic applications, including cutting stock problems [70], airline crew scheduling [12], political redistricting [118], multicommodity flow [3] and vehicle routing [163].

Branch and price is the application of column generation throughout the branch-and-bound tree to solve a large-scale integer program. The idea of combining the technique of column generation with an LP-based branch-and-bound algorithm was first suggested by

Desrosiers et al. [53] to solve a vehicle routing problem with time windows. For further details on column generation, and branch and price, the reader is referred to Barnhart et al. [14], and Lübbecke and Desrosiers [110].

2.2 MULTI-OBJECTIVE PROGRAMMING (MOP)

Multi-objective programming deals with decision problems characterized by multiple and possibly conflicting objective functions that are to be optimized over a feasible set of decisions [63]. Multi-objective programming constitutes a vast area of research with numerous modeling and solution techniques. In this section, we briefly introduce the general framework on multi-objective programming, and emphasize its applications in health care and the methodologies that are related to the techniques we utilize throughout the dissertation. From a methodological point of view, we especially focus on multi-objective integer programs. Much of this section follows the outline and notation of Ehrgott [59], and Ehrgott and Wiecek [63].

A general formulation for a multi-objective program (MOP) is given by

$$\min Cx \tag{2.1a}$$

$$\text{subject to } x \in X, \tag{2.1b}$$

where $X \subset \mathbb{R}^n$ denotes a feasible set. Let C be a $p \times n$ objective function matrix, i.e., $C : \mathbb{R}^n \to \mathbb{R}^p$ is a linear objective function. Suppose that X can be written as a set of linear constraints: $X = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\}$, where $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$. Let c_k denote the k^{th} row of C so that $y_k = c_k$ is the k^{th} objective value.

A feasible solution $x^* \in X$ is *efficient*, or *Pareto optimal* if there is no $x \in X$ such that $Cx \leq Cx^*$, and $C_i x < C_i x^*$ for at least one row C_i. Let $Y := CX := \{Cx : x \in X\} \subset \mathbb{R}^p$ denote the image of X under C. We say $y^* = Cx^*$ is *nondominated* if x^* is efficient. The set of all efficient solutions to MOP is $X_E \subset X$ and the set of nondominated points in criterion space is $Y_N \subset Y$. Solving MOP usually means finding X_E or Y_N.

Solution approaches for MOPs usually incorporate *scalarization*, which involves formulating a single-objective problem by means of a scalarizing function [63]. The scalarizing function is typically a function on the objective functions of the MOP. A popular solution approach is the *weighted sum* approach in which a weighted sum of the objective functions is minimized:

$$\min \sum_{k=1}^{p} \lambda_k c_k x \tag{2.2a}$$

$$\text{subject to } x \in X, \tag{2.2b}$$

where $\lambda_k \geq 0$ is the weight of the k^{th} objective function, for $k = 1, \ldots, p$. The most important property of the weighted sum approach is that (2.2) has the same computational complexity as the single objective version of MOP (2.1) [59]. The weighted sum approach also has variants such as the *weighted* t^{th} *power* approach, in which a weighted sum of the objective functions taken to the t^{th} power is minimized, and the *weighted quadratic* approach where a quadratic function of the objective functions is minimized.

Another popular scalarization approach is the $\varepsilon-$*constraint* approach, in which one objective function is retained while all other objective functions are replaced by new constraints. For instance, if the j^{th} objective function is retained for minimization and all other $p-1$ are turned into constraints, the j^{th} $\varepsilon-$constraint problem can be formulated as

$$\min c_j x \tag{2.3a}$$

$$\text{subject to } c_k x \leq \varepsilon_k, \qquad k \neq j, \tag{2.3b}$$

$$x \in X. \tag{2.3c}$$

For general results on the $\varepsilon-$constraint method, see [32]. Two important results are as follows: an optimal solution x^* to (2.3) is weakly efficient, i.e., there is no $x \in X$ such that $Cx < Cx^*$. Furthermore, if the optimal solution to (2.3) is unique, it is also efficient [32, 59].

Solution techniques to approximate the Pareto set usually involve iterative methods that incorporate a scalarization technique of choice to generate Pareto optimal points during the solution procedure. For a recent survey of these techniques, the reader is referred to [63].

Two other well known multi-objective optimization methodologies, which are not scalarization procedures, are *goal programming (GP)* and the *analytic hierarchy process (AHP)*. In GP, the decision maker is interested in achieving a desirable goal or target established for the objective functions of the MOP. GP was introduced by Charnes et al. [34]. The technique was formalized by Charnes and Cooper [33], and they introduced the term "goal programming". GP has been applied to many decisions, particularly in the last three decades [162]. AHP, which was introduced by Saaty [149], is a technique particularly suitable for complex decisions that involve the comparison of decision elements that are difficult to quantify. It involves building a hierarchy, or ranking, of decision elements and then making comparisons between each possible pair in each cluster, as a matrix. This gives a weighting for each element within a cluster, or level of the hierarchy, and also a consistency ratio.

2.2.1 Multi-objective Integer Programming (MOIP)

In Chapter 3, we construct a multi-objective integer programming framework to analyze the liver allocation hierarchy under two objectives: maximizing efficiency and maximizing equity. We use an iterative procedure built around the $\varepsilon-$constraint scalarization technique to build the Pareto efficient frontier. Thus, we are particularly interested in methodologies for multi-objective integer programming. Modifying the notation used for MOP, we can simply define a general multi-objective integer programming (MOIP) formulation by (2.1), where the feasible set $X \subset \mathbb{Z}^n$ is finite, C is a $p \times n$ objective function matrix with integer coefficients c_{ki} for $k = 1, \ldots, p, i = 1, \ldots, n$, i.e., $C : \mathbb{Z}^n \to \mathbb{Z}^p$ is a linear objective function, and X can be written as a set of linear constraints: $X = \{x \in \mathbb{Z}^n : Ax = b, x \geq 0\}$, where $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$.

Although the weighted sum approach is an exact method for multi-objective linear programs, the discrete structure of MOIP destroys this property. Hence, there usually exists efficient solutions, called *nonsupported* efficient solutions, that are not optimal for any weighted sum of the objectives. However, the exactness of the $\varepsilon-$constraint technique still holds for the integer case [58, 59, 60]. As we later discuss in Section 2.4, this shows that our methods in Chapter 3 that build parametric integer programs based on the $\varepsilon-$constraint approach, dominate the weighted sum approaches utilized by Stahl et al. [157] and Kong [96].

Our parametric integer programming models in Chapter 3 are solved using branch-and-price. Thus, we utilize column generation within a multi-objective integer programming framework. To the best of our knowledge, Ehrgott and Tind [62] provided the only other study that uses column generation while solving a multi-objective integer program. In this paper, they introduced a way to combine the $\varepsilon-$constraint method with a branch-and-bound approach, and discussed how the method can also be applied using the elastic constraint scalarization method of Ehrgott and Ryan [61].

For more details on MOIP we refer to [58, 60, 63].

2.2.2 MOP and MOIP Applications in Health Care

Health-care topics that have been studied under a multi-objective decision-making framework include nurse scheduling [10, 20, 66, 124, 130, 131, 166], tour planning for mobile health care facilities [54], capital budgeting in hospitals [92], health care planning [65, 142], location and size of medical departments in a hospital network [158], radiotherapy treatment design [75], priority setting in health policy [25, 129], information resource planning for health-care systems [102], balancing policy in long-term care [35], community-based residential care [127], design of ambulance services [11, 189], diet planning [146], blood center planning [91], and time allocation in pharmacies [69].

2.3 STOCHASTIC PROGRAMMING (SP)

Stochastic programs are mathematical programs where model parameters are subject to uncertainty. In this section, we will focus on the most widely studied stochastic programming models, i.e., two-stage stochastic linear programs. For more information on the field stochastic programming, we refer to Birge and Louveaux [22].

In a conventional two-stage stochastic linear program, a first-stage LP must be solved before all of the problem parameters are known with certainty. Once the uncertainty is realized, the decision maker then solves a second linear program, known as a *recourse* problem, that considers the solution to the first LP as well as the outcome of a random event. The objective is to minimize the first-stage cost plus the expected second-stage cost.

Let n_i be the number of columns in decision stage i for $i = 1, 2$. Let A be a real-valued matrix of size $m_1 \times n_1$, and b be a vector in $\mathbb{R}^{m_1}$. Let ξ be a discrete random variable describing the uncertain parameters, and let Ξ be the finite support of ξ. For $k = 1, \ldots, K = |\Xi|$, let ξ^k describe the k^{th} element in Ξ, called a *scenario*, and let p^k be the probability that scenario ξ^k is realized. For each scenario $\xi^k \in \Xi$, let $\{y(\xi^k) | T(\xi^k)x + W(\xi^k)y(\xi^k) \geq h(\xi^k), y(\xi^k) \geq 0\}$ be the set of feasible solutions for the second stage, where the *technology matrix*, $T(\xi^k)$, and the *recourse matrix*, $W(\xi^k)$, are of sizes $m_2 \times n_1$, $m_2 \times n_2$, respectively.

Beale [15] and Dantzig [41] formulated a two-sage stochastic linear program as follows:

$$\min \; c^T x + \mathbb{E}_\xi[d(\xi)^T y(\xi)] \tag{2.4a}$$

$$\text{subject to} \quad Ax \geq b, \tag{2.4b}$$

$$T(\xi^k)x + W(\xi^k)y(\xi^k) \geq h(\xi^k), \qquad \text{for } k = 1, \ldots, K, \tag{2.4c}$$

$$x \geq 0, \tag{2.4d}$$

$$y(\xi^k) \geq 0, \qquad \text{for } k = 1, \ldots, K, \tag{2.4e}$$

where c is a known vector in $\mathbb{R}^{n_1}$, and for each scenario, $d(\xi^k) \in \mathbb{R}^{n_2}$, $h(\xi^k) \in \mathbb{R}^{m_2}$. A scenario vector, $\xi^k = (d(\xi^k)^T, h(\xi^k)^T, T(\xi^k), W(\xi^k))$, represents the stochastic components of the problem, for $k = 1, \ldots, K$. For a given x, the second-stage subproblem decomposes into K independent subproblems, one for each scenario.

The extensive form (2.4) is equivalent to the *deterministic equivalent program*:

$$\begin{aligned} \min\ & c^T x + \mathcal{Q}(x) && (2.5)\\ \text{subject to}\quad & Ax \geq b, \\ & x \geq 0, \end{aligned}$$

where $\mathcal{Q}(x)$, the *expected recourse function*, is defined as

$$\mathcal{Q}(x) = \mathbb{E}_{\xi} Q(x, \xi^k)), \tag{2.6}$$

and for every scenario ξ^k,

$$\begin{aligned} Q(x, \xi^k) = \min\ & d(\xi^k)^T y && (2.7a)\\ \text{subject to}\quad & W(\xi^k) y \geq h(\xi^k) - T(\xi^k) x, && (2.7b)\\ & y \geq 0. && (2.7c) \end{aligned}$$

The most common algorithm used for solving these types of problems is the *L-shaped method* of Van Slyke and Wets [174], a variant of Benders decomposition [17] that decomposes the problem into first-stage (or *master*) variables and second-stage (or *subproblem*) variables. The L-shaped method follows an iterative procedure where it repeatedly solves a restricted master problem to find first-stage solutions, and then it solves the second-stage problems, given the first-stage solution. The second-stage solutions are used to find optimality and feasibility cuts for the restricted master problem. Using Minkowski's finite basis theorem [121, 128], the second-stage dual polyhedron is represented as a convex combination of its extreme points and a nonnegative linear combination of its extreme rays. The cuts correspond to these extreme points and rays.

In this dissertation, we introduce a general method that combines Dantzig-Wolfe decomposition and the L-shaped method to solve large-scale two-stage stochastic linear programs. This method generates columns as needed for the master problem of the L-Shaped method with respect to the first-stage constraints *and* current feasibility and optimality cuts. It also adds feasibility and optimality cuts generated from the second-stage subproblems based on the current columns in the master problem. In other words, our method has a restricted

master problem (RMP) where second-stage variables and first-stage easy constraints are replaced by cuts and columns, respectively. We also discuss several variants of the algorithm and perform computational tests on their performances. Although column generation and stochastic programming are both popular topics in mathematical programming, there is very little research that combines both techniques. Lulli and Sen [111] used a branch-and-price algorithm to solve a multistage stochastic integer programming problem. Their technique applied the Dantzig-Wolfe decomposition principle based on the convexification of the integer scenario polyhedra and the generated columns represented extreme points of the scenario polyhedra.

Since the L-shaped method is based on Benders decomposition, the general method we introduce can be regarded as a combined Benders and Dantzig-Wolfe decomposition scheme. Cross-decomposition methods also combine Benders and Dantzig-Wolfe decomposition [76, 77, 78, 79, 173]. However, cross decomposition is fundamentally different from our approach in that it decomposes both the primal problem and the Lagrangian dual problem into a restricted master problem and a subproblem. Both subproblems are solved at every iteration. On certain major iterations, one of the two master problems is solved to update its corresponding subproblem.

2.3.1 SP Applications in Health Care

Stochastic programming literature in health care is modest in terms of volume when compared to medical applications of deterministic linear/integer or multi-objective programs. In this section, we discuss a few representative studies that use stochastic programming to model health care systems.

Beraldi et al. [19] developed a stochastic programming model with probabilistic constraints to locate service sites and determine the number of emergency vehicles assigned to each site, with the aim of achieving reliable service levels while minimizing the overall costs.

Kao and Queyranne [90] used stochastic programming models to determine budgets for nursing workforce requirements in a hospital.

Punnakitikashem et al. [138] developed a two-stage stochastic integer programming model for assigning nurses to patients under uncertainty with the objective of minimizing excess workload for nurses.

Martel and Ouellet [114] modeled the problem of allocating resources among partially interchangeable activities as a stochastic program with complete recourse. They reduced the problem to a deterministic convex allocation problem through parametric programming. They also provided an application of the model studied in the allocation of the budget of a nursing unit in a hospital to registered nurses.

Lamiri et al. [99] introduced a stochastic model for operating room planning with two types of demand for surgery: elective surgery, which can be planned ahead of time, and emergency surgery. Their model assigns elective cases to different periods of a planning horizon with the objective of minimizing total costs of operating rooms.

Sapountzis [151] formulated the problem of allocating units of blood from a regional blood transfusion center to hospitals as a stochastic programming problem. He showed that his formulation reduces to a deterministic linear program.

2.4 ORGAN ALLOCATION AND TRANSPLANTATION

In this section, we focus our attention to previous studies on organ allocation and transportation. This is by no means a complete and exhaustive list of studies focusing on organ allocation systems, but is an attempt to introduce a representative set of publications to display the growing interest in this area.

For the past two decades, operations research applications on organ transplantation have gained significant momentum. Generally, the studies in this area have focused on recipient-donor matching for organ transplantation to maximize a reward function. These studies have emphasized either the patient's perspective for deciding whether to accept an organ offer or not, or the perspective of a centralized decision maker representing the common interests of the society. Recently, joint perspectives of both the patient and society have also been researched. Alagoz et al. [8] provided a comprehensive survey of operations research applications related to organ allocation and transplantation.

David and Yechiali [45] were among the first to model the perspective of a patient while deciding whether to accept or reject a liver offer. The authors presented a time-dependent control-limit policy under the assumption that organs arrive only at fixed time intervals. They also modeled the organ arrival as a renewal process and assumed that patients' health states are always deteriorating. However, the authors did not model the system under the actual matching criteria, and did not consider the waiting list.

Ahn and Hornberger [2] adopted a Markov Decision Process (MDP) framework to model the decision regarding which kidneys maximize the duration and quality of life for a patient. Hornberger and Ahn [80] used an MDP model to design kidney acceptance policies with an explicit incorporation of the preferences of patients. They also modeled the waiting list and estimated one-year survival rates after accepting an offer an receiving a transplant. In both studies, patient health states are oversimplified and do not model physiology. For a comprehensive review on MDPs, the reader is referred to Puterman [139].

Howard [81] introduced a decision model where a cadaveric organ is accepted or rejected depending on the patient's health state by her transplant surgeon. He also discussed the reasons why a surgeon may chose to reject an organ offer and provided statistical evidence.

More recently, Alagoz et al. [4] introduced an MDP model to determine optimal acceptance/rejection decisions for ESLD patients waiting for living-donor liver transplants. They showed the existence of optimal control-limit policies under mild conditions, and used the model of Alagoz et al. [7] to estimate the progression of ESLD. In Alagoz et al. [6], the authors investigated optimal acceptance/rejection policies for patients waiting for cadaveric liver donations under an implicit representation of the waiting list. Alagoz et al. [5] extended the MDP framework to the case where an ESLD patient can choose among cadaveric- and living-donor transplants.

Previous studies focusing on the societal perspective usually involve multiple potential recipients and organ offers under a stochastic modeling framework to capture the arrival and status changes. Righter [143] developed a stochastic sequential assignment problem [52] and provided structural properties for optimal policies. David and Yechiali [46] constructed an infinite-horizon model in which potential recipients and organ offers arrive simultaneously. In a later study, David and Yechiali [47] extended the modeling framework to incorporate the random arrival of organs. They considered special cases where organ and patient numbers are equal, there are more organs than patients, and there are more patients than organs. They derived optimal matching policies to maximize the total discounter reward under a static representation of patient health states where the waiting list is ignored.

Zenios et al. [187] constructed a Monte Carlo simulation model to compare several kidney allocation policies. They used quality-adjusted life expectancy and survival rates as performance measures. The authors also considered the notion of equity as measured by the average waiting time for transplants for different ethnic groups. They showed that evidence-based kidney allocation policies could potentially improve both efficiency and equity when compared to current UNOS policies. However, their study was limited to the designated service area of a single OPO. Zenios et al. [186] studied the problem of finding a kidney allocation strategy to maximize total quality-adjusted life years and minimize two inequity measures. The two inequity measures considered were the amount of access to organs across different groups and inequity in waiting times. The authors built a deterministic fluid model that did not lend itself to a closed-form solution. They employed a heuristic dynamic index policy.

Zenios [184] developed a queueing model with different classes of patients and organs, where patient death is modeled as reneging. He considered focused on randomized kidney allocation policies and derived closed-form asymptotic expressions for the stationary waiting time, stationary waiting time until transplantation, and proportion of patients who receive transplantation for each patient class. In a later study, Zenios [185] modeled the mix of direct and indirect kidney exchanges in order to maximize the expected total discounted quality-adjusted life years of the recipients in participating pairs. He developed a double-ended queueing model for an exchange system with two types of donor-candidate pairs, and derived an optimal dynamic exchange policy using Brownian approximation. He discussed a few design principles that would increase likelihood of the success of an exchange program.

Recent literature emphasizing the joint perspective of the patient and society include [159, 160, 161]. Su and Zenios [159] developed an M/M/1 queue with homogeneous patients and exponential reneging to model patient choice in the kidney transplant waiting system. Organ transplants are captured in the model through the service process, and each service instance has an associated variable reward that represents transplant quality. The patients have the option to refuse an organ. The authors tested different queueing strategies using data clinical data. Su and Zenios [160] modeled the problem of allocating kidneys to recipients when recipients have a right to refuse organ offers using a stochastic sequential assignment problem. They solved two variants of the model for the cases where patients cannot reject organ offers (and the numbers of offers and patients are equal) and when they have the right to do so. In both problems, they aimed to maximize the total reward which is a function of kidney and patient types. They derived structural properties for both cases, and measured the effects of patient autonomy on acceptance/rejection rates. More recently, Su and Zenios [161] developed a mechanism design model to explore the effect of information asymmetry in the kidney allocation system. In their model, there are different queues corresponding to different potential recipient types, and patients join queues to maximize their utility. Candidate types are only observed by the candidates, and each candidate chooses the queue to join by reporting a type. Kidneys have heterogeneous types, and each kidney is assigned to one of the queues depending on its type. They calculated utility using fluid approximations, and used two alternative social welfare functions: aggregate utility (em-

phasizing efficiency) and minimum utility across all candidates (emphasizing equity). They solved the problem under both objective functions in order to divide the organ supply among the different queues to maximize social welfare.

A number of discrete-event simulation models have also been developed in order to facilitate policy design and evaluation for the U.S. liver allocation and transplantation system. UNOS Liver Allocation Model (ULAM) was developed by UNOS and Pritsker Corporation to model the process of patient listing, organ donations and matching policies [137]. Another simulation model was developed by the CONSAD Corporation [40]. This model assumes that all patients are registered in a single national waiting list. More recently, Shechter et al. [155] built a discrete-event simulation model for ESLD and the liver allocation mechanism. This model also captures the regional effects of organ allocation and is able to simulate different regional configurations based on user inputs.

2.4.1 Previous Studies on Transplantation Region Design

To the best of our knowledge, the problem of optimally reorganizing regions for liver transplantation was first addressed by Stahl et al. [157]. They formulated a set-partitioning model to address the problem of optimizing the regional designs for liver allocation across the United States, with binary variables representing different regions. They solved the problem using an explicit enumeration of regions, where the set of regions was limited to contiguous sets of no more than 9 OPOs. They also addressed the issue of geographic equity using a multi-objective approach where they had a weighted equity component in the objective. They aimed to maximize the minimum equity measure of intra-regional transplant rate. By performing sensitivity analysis on the weight component they demonstrated the tradeoff between two objectives of maximizing efficiency and equity for a set of different objective weights on the equity measure. Their regional efficiency and equity benefit estimates did not include any component to capture the likelihood of organ allocation between OPO pairs but were rather in terms of rates computed over the number of patients listed in different OPOs.

The models in [157] were extended by Kong [96] and Kong et al. [97] by employing a Branch-and-Price approach that enabled region generation on-the-fly and lifted the hard

limits on the number of OPOs in each region. Furthermore, in these studies, a more accurate estimate of the efficiency benefit that also captured the likelihood of liver allocation between OPO pairs was utilized. Kong [96] revisited the multi-objective approach for balancing allocation efficiency and equity with minor adjustments on the equity benefit estimate. He also solved the combined efficiency and equity problem using an explicit enumeration of regions, with different weights on the equity component.

As we discussed in Section 1.3, in this dissertation, we introduce approaches that extend all previous work [96, 97, 157] on balancing efficiency and equity in the liver allocation system. We provide exact methods to enumerate nondominated efficient frontier for the two objectives of maximizing efficiency and equity. Stahl et al. [157] and Kong [96] used a set of predetermined objective weights throughout their analysis instead, which is known to be suboptimal [58, 59, 60], as we pointed out in Section 2.2. Furthermore, we solve the multi-objective optimization problem within a branch-and-price setting, which lifts the limit on the number of OPOs in regions that previous studies had due to their explicit enumeration based techniques. We also use a refined equity measure that also incorporates the likelihood of organ-patient matches between OPO pairs.

In addition to these, we also lift the steady-state assumption that was common in all previous studies on the design of regions in the U.S. for the liver transplantation mechanism [96, 97, 157]. In this dissertation, we construct two stochastic programming models that capture the stochastic behavior of the national waiting list for livers. We also adopt a different objective than all other previous studies on the region design problem [96, 97, 157] in these chapters. To the best of our knowledge, this dissertation is the first study that aims to optimize the expected life-days gained with liver transplants through region redesign.

3.0 BALANCING EFFICIENCY AND EQUITY IN THE LIVER ALLOCATION HIERARCHY

3.1 INTRODUCTION

In this chapter, we introduce approaches that extend the literature [96, 97, 157] on balancing efficiency and equity in the liver allocation system. We introduce exact methods that enable a more comprehensive comparison and balancing of the two possibly conflicting objectives of maximizing efficiency and equity. Under a multi-objective optimization framework, we use two parametric integer programming models. The first maximizes the efficiency of the allocation mechanism by considering only those region designs for which OPOs have sufficient access to livers, while the second maximizes the equity measure across all OPOs subject to minimum total efficiency. We solve many submodels using branch and price repeatedly within an iterative process to approximate the efficient frontier of Pareto optimal solutions.

The solution methods that we discuss in this chapter can give the exact efficient frontier within any degree of accuracy. We use iterative methods in which the construction of the efficient frontier is guided by sensitivity analysis techniques to aid in the selection of parameters for ε-constraint subproblems. Previous studies [96, 97, 157] used a set of predetermined objective weights throughout their analysis instead, which is known to be suboptimal since the weighted sum approach cannot be used to enumerate all efficient solutions for multi-objective integer programs [58, 59, 60]. Furthermore, the solution approach that was common in previous studies on balancing efficiency and equity was the explicit enumeration of a set of contiguous regions with no more than 9 OPOs as candidate regions. We employ a branch-and-price approach that lifts the limit on the number of OPOs in regions.

The chapter is structured as follows: in Section 3.2, we introduce two parametric integer programming models to analyze the two objectives of maximizing efficiency and maximizing a minimum equity measure across all OPOs. We discuss our branch-and-price approaches and derive pricing problems that will aid us in generating favorable regions "on-the-fly". Section 3.3 focuses on the algorithmic details in the branch-and-price implementation, and constructs two algorithms to approximate the efficient frontier of the system. In Section 3.4, we present and discuss our numerical results. And finally, in Section 3.5 we summarize our findings for this chapter.

3.2 MATHEMATICAL MODELS

As we discussed in Section 1.2, we focus on MELD patients on the waiting list at the regional matching phase, and ignore Status 1 patients and the national matching step in this chapter. Status 1 patients only make up less than 0.1% of the ESLD patient population and the percentage of transplants at the national level is only around 5% [169]. Furthermore, incorporating these complicates the model structure considerably and disrupts the computational tractability of the problem. All studies on region design [96, 97, 157] utilized the same simplification and focused on the regional organ allocation step for MELD patients, i.e., Step 4 in Figure 1.3.

We make use of two parametric integer programming models throughout our analysis. In Section 3.2.1, we introduce a parametric integer program that maximizes the efficiency of the allocation mechanism considering only those region designs in which all OPOs have sufficient access to organs. And later in Section 3.2.2, we introduce our second model which maximizes a minimum equity measure throughout the nation subject to a minimum total efficiency parameter.

3.2.1 Mathematical Model for Maximizing Efficiency

The *efficiency* model maximizes the efficiency of the liver allocation system, as measured by the total national expected benefit of the region design, subject to a minimum equity level for each OPO. The equity level captures the access of the population to organs in various geographic locations throughout the nation.

We extend the notation developed in [97]. Let I denote the set of OPOs across the U.S., R be the set of all potential regions of OPOs, and I_r denote the set of OPOs in region $r, \forall r \in R$. Also, define c_r as the regional efficiency benefit of region r, and f_{ir} as the expected intra-regional equity benefit of region r for OPO i, given $i \in I_r$. Let λ represent the smallest desired geographic equity measure across OPOs. Let the binary coefficient $a_{ir} = 1$, if OPO i is in region r, and $a_{ir} = 0$ otherwise. Finally, let x_r be a binary variable such that $x_r = 1$ if region r is selected in the regional configuration and $x_r = 0$ otherwise, for each $r \in R$.

Note that the decision variables and parameters introduced depend on the set R, which is exponential in size and impractical to enumerate explicitly. For instance, the number of contiguous regions with no more than 8 OPOs is around 3.44×10^5, whereas there are 1.27×10^6 contiguous regions with up to 9 OPOs [96]. Therefore, the problem is solved within a branch-and-price framework, by generating promising regions on-the-fly throughout the branch-and-bound tree. The *restricted* set of regions, which consists of the regions generated throughout the flow of the algorithm will be denoted by R'. Note that $R' \subseteq R$ and that all of the definitions above are valid on the R' domain. Then, we have the following restricted parametric mathematical program given parameter λ:

$$\eta(\lambda) = \max \sum_{r \in R'} c_r x_r \tag{3.1a}$$

$$\text{subject to} \sum_{r \in R'} a_{ir} x_r = 1, \qquad \forall i \in I, \tag{3.1b}$$

$$\sum_{r \in R'} f_{ir} x_r \geq \lambda, \qquad \forall i \in I, \tag{3.1c}$$

$$x_r \in \{0, 1\}, \qquad \forall r \in R'.$$

The objective function (3.1a) maximizes the expected benefit of liver transplantation. The set-partitioning constraint set (3.1b) ensures that each OPO is contained within one

single region in the regional configuration. Constraint set (3.1c) requires the minimum equity measure to be satisfied across all OPOs. Although $\eta(\lambda)$ also depends on the restricted set of regions, R', we drop R' from notation for ease of exposition.

Estimating the Regional Efficiency and Equity Benefits

The regional efficiency benefit is based on two components: allocation likelihood (i.e., the likelihood of organ-patient matches between different OPOs) and organ viability (i.e., the organ's quality loss due to cold ischemia time).

Kong et al. [97] utilized two important assumptions that come into play while analyzing allocation likelihood. We keep these assumptions in our models as well.

Assumption 3.1. *[97] The allocation process is in steady state.*

Assumption 3.2. *[97] The allocation likelihood depends only on the donor OPO, the recipient OPO, and the potential region containing the two OPOs.*

These assumptions enable a closed-form analysis of organ flows between OPOs. The allocation mechanism is simplified through aggregating donors and patients, and allocation of organs to recipients is modeled based on a macro-level *proportional allocation* scheme. Thus, proportional allocation assumes that the flow of organs between two OPOs in any potential region would be proportional to the flow of organs between the same OPOs under a single national waiting list.

When an organ is matched to a patient, the benefit of the transplant is affected by the decay of organ quality, which directly affects any outcome measure that accounts for viability adjustment, such as post-transplant survival rates. The organs lose viability due to CIT, and the main contributor to the duration the organ spends outside of the human body is the distance it has to be transported. The following assumption from [96] deals with the viability loss.

Assumption 3.3. *[96] The viability-adjusted outcome for each transplant is only dependent upon the locations of donor and recipient OPOs.*

Here we list additional parameters to be used in computing intra-regional efficiency and equity benefits. We use the notation from [96] and [97]. Define o_i to be the number of

organs procured at OPO $i \in I$ over a period of time and p_i to be the number of patients who register on the waiting list at OPO $i \in I$. We assume $p_i > 0, \forall i \in I$. Let α_{ij} be the viability-adjustment multiplier for a transplant between donor OPO i and recipient OPO j, which is directly affected by the organ transport distance between the two OPO service areas. This multiplier captures the quality of the transplant after adjusting for the viability loss and can be interpreted as follows: a decision maker would be indifferent between the benefit gained by transplanting an organ procured at OPO i to a patient at OPO j, and α_{ij} times the benefit of transplanting the same organ to the same patient at OPO i [96]. Two more parameters are needed to capture the allocation likelihood between OPO pairs: let l_{ij} and l_i^0 denote the *pure distribution likelihood* from donor OPO i to recipient OPO j, and the *pure national flow likelihood* from donor OPO i, respectively. The *pure distribution likelihood* is the likelihood of organ distribution between OPO pairs under a single national waiting list. The expected likelihood of organ wastage and organ flow to all OPOs outside of any potential region for the donor OPO is called the *pure national flow likelihood*. The reader is referred to [96, 97] for further details.

Now we are ready to describe how to estimate c_r for any possible region $r \in R$. Given two OPOs $i \neq j$ in region r, let z_{ij} denote the allocation likelihood from OPO i to OPO j. Although z_{ij} is region-dependent we omit the region index r from notation for ease of exposition. Then, for two OPOs i and j in region r, z_{ij} can be calculated as follows:

$$z_{ij} = \frac{l_{ij}}{\sum_{k \in I_r \setminus \{i\}} l_{ik} + l_i^0}. \tag{3.2}$$

Hence, the *expected viability-adjusted number of transplants* between donor OPO i and recipient OPO j will be equal to $o_i z_{ij} \alpha_{ij}$. Adding up this benefit for all OPOs in region r gives the intra-regional efficiency benefit for region r, namely c_r:

$$c_r = \sum_{i \in I_r} \sum_{j \in I_r \setminus \{i\}} o_i \,.\, z_{ij} \,.\, \alpha_{ij} = \sum_{i \in I_r} \sum_{j \in I_r \setminus \{i\}} o_i \,.\, \frac{l_{ij}}{\sum_{k \in I_r \setminus \{i\}} l_{ik} + l_i^0} \,.\, \alpha_{ij}. \tag{3.3}$$

This, in fact, is the same definition of intra-regional transplant benefits that Kong [96, 97] used. Stahl et al. [157] did not account explicitly for the allocation likelihood of organs between OPO pairs but assumed the flow of organs was proportional to the size of patient populations in the OPOs of a region.

We define the expected intra-regional equity benefit of region r for OPO i, i.e., f_{ir}, as the *rate of likelihood-adjusted intra-regional transplants per-patient* for OPO i given $i \in I_r$. Following our definition, f_{ir} can be calculated as follows:

$$f_{ir} = \begin{cases} \sum_{j \in I_r \setminus \{i\}} \left(\frac{o_j}{p_i}\right) \cdot z_{ji} \cdot \alpha_{ji}, & \text{if OPO } i \in \text{region } r, \\ 0, & \text{otherwise.} \end{cases} \tag{3.4}$$

Thus, for OPO i in region r, $f_{ir} = \sum_{j \in I_r \setminus \{i\}} \left(\frac{o_j}{p_i}\right) \cdot \frac{l_{ji}}{\sum_{k \in I_r \setminus \{j\}} l_{jk} + l_j^0} \cdot \alpha_{ji}$.

Our definition of the intra-regional equity benefit per OPO differs from that of Kong [96] and Stahl et al. [157]. Both previous studies neglected the allocation likelihood among OPOs when deriving the equity measure. The estimates they used are defined as *intra-regional transplant rates* for an OPO in a certain region. Stahl et al. [157], defined equity as

$$\sum_{j \in I_r \setminus \{i\}} o_j \cdot \frac{1}{(\sum_{k \in I_r} p_k) - p_j} \cdot \alpha_{ji},$$

while Kong [96] defined it as

$$\sum_{j \in I_r \setminus \{i\}} o_j \cdot \frac{1}{\sum_{k \in I_r} p_k} \cdot \alpha_{ji}.$$

Neither of these definitions consider the effects of allocation likelihood between OPO pairs in the regions. Instead, they compute hypothetical rates of intra-regional transplants based on patient populations without incorporating the probabilistic organ flows between OPOs that compose a region. However, we include an additional allocation likelihood component adjusting the rate of intra-regional transplants received by an OPO by the likelihood of organ flow into that OPO from its own region.

Generating Regions for the Efficiency Model

Denote the dual variables associated with constraints (3.1b) and (3.1c) by π and σ, respectively. Then the reduced cost of a potential region $r \in R$ can be calculated as follows:

$$\bar{c}_r = c_r - \sum_{i \in I} a_{ir}\pi_i - \sum_{i \in I_r} f_{ir}\sigma_i \tag{3.5a}$$

$$= \sum_{i \in I_r} \sum_{j \in I_r \setminus \{i\}} o_i z_{ij} \alpha_{ij} - \sum_{i \in I_r} \pi_i - \sum_{i \in I_r} \sum_{j \in I_r \setminus \{i\}} \left(\frac{o_j}{p_i}\right) z_{ji} \alpha_{ji} \sigma_i \tag{3.5b}$$

$$= \sum_{i \in I_r} \sum_{j \in I_r \setminus \{i\}} \left[o_i \alpha_{ij} \left(1 - \frac{\sigma_j}{p_j}\right) z_{ij} \right] - \sum_{i \in I_r} \pi_i. \tag{3.5c}$$

We modify Kong's pricing problem [96] to include the effects of the equity rows and employ the following MIP within our branch-and-price framework to generate promising regions throughout the branch-and-bound tree.

$$RPP_\eta(\pi, \sigma, \lambda) = \max \sum_{i \in I} \sum_{j \in I \setminus \{i\}} \left[o_i \alpha_{ij} \left(1 - \frac{\sigma_j}{p_j}\right) z_{ij} \right] - \sum_{i \in I} \pi_i y_i \tag{3.6a}$$

$$\text{subject to} \quad \sum_{j \in I \setminus \{i\}} z_{ij} + z_i^0 = y_i, \qquad \forall i \in I, \tag{3.6b}$$

$$z_{ij} \leq y_j, \qquad \forall i, j \in I, i \neq j, \tag{3.6c}$$

$$l_{ik} z_{ij} \leq l_{ij} z_{ik} + l_{ik}(1 - w_{jk}), \qquad \forall i, j, k \in I, i \neq j, k, j < k, \tag{3.6d}$$

$$l_{ij} z_{ik} \leq l_{ik} z_{ij} + l_{ij}(1 - w_{jk}), \qquad \forall i, j, k \in I, i \neq j, k, j < k, \tag{3.6e}$$

$$l_i^0 z_{ij} \leq l_{ij} z_i^0 + l_i^0(1 - w_{ij}), \qquad \forall i, j \in I, i < j, \tag{3.6f}$$

$$l_{ij} z_i^0 \leq l_i^0 z_{ij} + l_{ij}(1 - w_{ij}), \qquad \forall i, j \in I, i < j, \tag{3.6g}$$

$$l_j^0 z_{ji} \leq l_{ji} z_j^0 + l_i^0(1 - w_{ij}), \qquad \forall i, j \in I, i < j, \tag{3.6h}$$

$$l_{ji} z_j^0 \leq l_j^0 z_{ji} + l_{ij}(1 - w_{ij}), \qquad \forall i, j \in I, i < j, \tag{3.6i}$$

$$w_{ij} \geq y_i + y_j - 1, \qquad \forall i, j \in I, i < j, \tag{3.6j}$$

$$\sum_{j \in I \setminus \{i\}} \left(\frac{o_j \alpha_{ji}}{p_i}\right) z_{ji} \geq \lambda y_i, \qquad \forall i \in I, \tag{3.6k}$$

$$y_i \in \{0, 1\}, 0 \leq z_i^0 \leq 1, \forall i \in I, \tag{3.6l}$$

$$0 \leq z_{ij} \leq 1, \forall i, j \in I, i \neq j, 0 \leq w_{ij} \leq 1, \forall i, j \in I, i < j. \tag{3.6m}$$

The objective function (3.6a) of the above formulation maximizes the reduced cost of a region. Constraints (3.6b) force the livers procured at an OPO to be allocated at the regional level only if that particular OPO is chosen in the selected region. Constraint (3.6c) allows organs procured at OPO i to be allocated to OPO j only if OPO j is chosen in the region design.

In this pricing problem, variables w_{ij} help enforce proportional allocation requirements. For instance, if OPOs i, j and k are chosen for a potential region, then by constraints (3.6j) and (3.6m), $w_{jk} = 1$. This forces constraints (3.6d) and (3.6e) to be satisfied with equality, and hence, proportional allocation between OPOs j and k is enforced. In a similar fashion, constraint sets (3.6f-3.6g) and (3.6h-3.6i) make sure that proportional allocation requirements between OPO pairs in the same region and the regional level are met.

Formulation (3.6) differs from Kong's pricing problem in two ways: the objective function (3.6a) is different due to differences in the master problem structure, and we have additional constraints (3.6k). Thus, Kong's pricing problem [96] is a special case of the one shown above, with $\lambda = 0, \sigma = 0$. Kong [96] proved that the pricing problem he formulated is NP-hard by reducing from the maximum facility location problem. Hence, (3.6) is also NP-hard.

3.2.2 Mathematical Model for Maximizing Equity

Let η denote the smallest allocation efficiency measure for the whole nation. Then, we have the following restricted parametric mathematical program over η:

$$\lambda(\eta) = \max \lambda \tag{3.7a}$$

$$\text{subject to} \sum_{r \in R'} a_{ir} x_r = 1, \qquad \forall i \in I, \tag{3.7b}$$

$$\sum_{r \in R'} f_{ir} x_r \geq \lambda, \qquad \forall i \in I, \tag{3.7c}$$

$$\sum_{r \in R'} c_r x_r \geq \eta, \tag{3.7d}$$

$$x_r \in \{0, 1\}, \qquad \forall r \in R',$$

$$\lambda \geq 0.$$

The objective function (3.7a) maximizes the minimum equity measure across all OPOs in the country. The set-partitioning constraints (3.7b) and equity constraints (3.7c) stay the same. Constraint (3.7d) ensures that the regional design fulfills the minimum allocation efficiency requirement. Note that the left-hand side of this constraint actually is the objective function of formulation (3.1), and the right-hand side of constraint set (3.1c) has been incorporated into the objective function in the equity formulation (3.7). Once again, although $\lambda(\eta)$ also depends on R' we drop R' from notation for ease of exposition.

Generating Regions for the Equity Model

Denote the dual variables associated with constraints (3.7b), (3.7c) and (3.7d) by π, σ and γ, respectively. Then the reduced cost of a potential region $r \in R$ can be calculated as follows:

$$\begin{aligned}
\bar{c}_r &= 0 - \sum_{i \in I} a_{ir}\pi_i - \sum_{i \in I_r} f_{ir}\sigma_i - c_r\gamma && \text{(3.8a)}\\
&= -\sum_{i \in I_r} \pi_i - \sum_{i \in I_r} \sum_{j \in I_r \setminus \{i\}} \left(\frac{o_j}{p_i}\right) z_{ji}\alpha_{ji}\sigma_i - \sum_{i \in I_r} \sum_{j \in I_r \setminus \{i\}} o_i z_{ij}\alpha_{ij}\gamma && \text{(3.8b)}\\
&= -\sum_{i \in I_r} \pi_i - \sum_{i \in I_r} \sum_{j \in I_r \setminus \{i\}} \left[o_i\alpha_{ij}\left(\gamma + \frac{\sigma_j}{p_j}\right) z_{ij}\right]. && \text{(3.8c)}
\end{aligned}$$

The pricing problem can be formulated as follows:

$$\begin{aligned}
RPP_\lambda(\pi, \sigma, \gamma) = \max & -\sum_{i \in I_r} \sum_{j \in I_r \setminus \{i\}} \left[o_i\alpha_{ij}\left(\gamma + \frac{\sigma_j}{p_j}\right) z_{ij}\right] - \sum_{i \in I} \pi_i y_i && \text{(3.9a)}\\
& \text{subject to } (3.6b) - (3.6j),\\
& \qquad\qquad (3.6l) - (3.6m).
\end{aligned}$$

The constraint structure of formulation (3.9) is identical to that of Kong's pricing problem [96]. The only difference between the two formulations is the objective function. The pricing problem utilized by Kong [96] is a special case of (3.9) with $\sigma = 0, \gamma = -1$. Hence, (3.9) is also NP-hard.

3.3 ALGORITHMIC APPROACHES

The goal is to find the efficient frontier of a system with two possibly conflicting objectives: maximizing efficiency versus maximizing equity. The models we employ are integer programs that are solved using a branch-and-price framework. This section focuses on describing how the efficient frontier will be approximated. First, we include some explanations about the computational approaches to the parametric integer programs that have been introduced in the previous section.

3.3.1 Computational Approaches for Solving the Efficiency and Equity Models

The efficiency of a branch-and-price algorithm, or more generally a column generation scheme, depends on the pricing strategy and how effectively the pricing problem can be solved. Column generation techniques usually utilize pricing problems that are "easy" to solve, in the sense that they are either well suited with special structure that can be efficiently exploited, or satisfy the integrality property. For instance, as we discuss in Chapter 5, the pricing problems of multi-commodity flow and cutting stock problems are shortest-path and integer knapsack problems, respectively. However, our pricing problems, (3.6) and (3.9), lack desirable properties that would enable fast solutions and are hard to solve. In order to alleviate the difficulty in solving these mixed integer programs, we utilize Kong's *Geographic Decomposition* approach [96, 97]. The main idea behind this technique is to create a number of smaller pricing problems, each covering different geographic areas of the country called *region covers*, instead of considering a very large pricing problem for the whole nation. We also use a multiple pricing scheme in which we insert *all* columns with favorable reduced costs found while optimizing the pricing problems, instead of just using the optimal solution.

3.3.1.1 Geographic Decomposition The number of constraints in the pricing problem for the efficiency model is $O(|I|^3)$, and the full pricing problem for the whole nation, which currently has $|I| = 58$, contains around 200,000 rows. When poor scalability properties of integer programming is taken into account, one would rather create a number of smaller pricing problems instead of solving a big one. This is the basic idea behind the Geographic Decomposition technique.

We start by designing *region covers*, which are geographic areas in the country that cover a number of OPOs. Each region cover has an associated pricing problem of its own, from which new favorable regions can be generated. Two important things about region covers are that we need a set of region covers that span the whole country, and these covers can overlap. In fact, having covers that overlap is essential because the model can recapture some potential regions that would be implicitly eliminated from consideration in case of non-overlapping region covers.

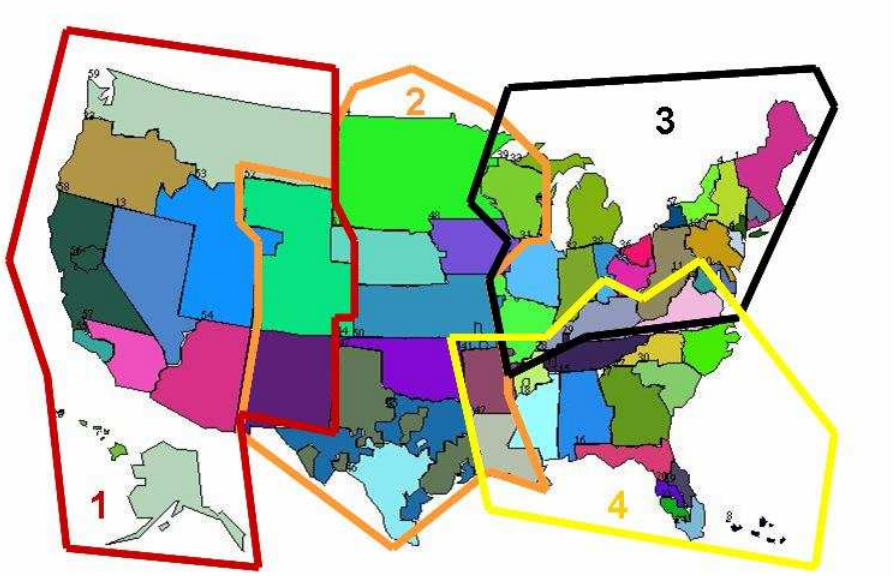


Figure 3.1: An example of geographic decomposition.

Figure 3.1 illustrates the idea of geographic decomposition through region covers. In this figure we have four overlapping region covers that span different portions of the country. Suppose we have a set of region covers denoted by $\mathcal{C}$. Let I_j denote the OPOs in region cover $j \in \mathcal{C}$. Let us denote the associated pricing problem for the efficiency and equity models for region cover $j \in \mathcal{C}$ by $RPP_\eta(\pi, \sigma, \lambda, I_j)$ and $RPP_\lambda(\pi, \sigma, \gamma, I_j)$, respectively. Replacing

the term I with I_j in (3.6) and in (3.9) gives the description of the efficiency and equity pricing problem for region cover $j \in \mathcal{C}$, respectively. Note that without using Geographic Decomposition, following the same notation, the pricing problems become $RPP_\eta(\pi, \sigma, \lambda, I)$ and $RPP_\lambda(\pi, \sigma, \gamma, I)$. Once a set of region covers $\mathcal{C}$ is designed, during every column generation subroutine, the associated pricing problems for every $j \in \mathcal{C}$ is solved until no favorable region is found in any $j \in \mathcal{C}$.

Although the basic idea of this approach is to create smaller subproblems, we would ideally want the region covers to be big enough so that it would be more likely to generate favorable regions that are potentially in the optimal basis of the set-partitioning master problem. Intuitively, the more and bigger region covers we have in set $\mathcal{C}$, the more likely we are to find a promising region. See Kong [96] for a detailed discussion on Geographic Decomposition.

3.3.1.2 Set-partitioning Branching We employ the branching strategy of Ryan and Foster [148] which has proven to be effective on set-partitioning type problems. Although they did not consider column generation applications of their branching strategy, it has been observed to be very useful in this context [115]. This branching rule is based on the following proposition:

Proposition 3.1. *[14, 115, 148] If A is a 0-1 matrix, and a basic solution to $Ax = 1$ is fractional, i.e., at least one of the components of x is fractional, then there exist two rows s and t of the master problem such that*

$$0 < \sum_{k:a_{sk}=1,a_{tk}=1} x_k < 1. \tag{3.10}$$

A pair of rows, s and t, that satisfies (3.10) gives the following pair of branching constraints:

$$\sum_{k:a_{sk}=1,a_{tk}=1} x_k = 1, \qquad \text{on one branch, and} \tag{3.11a}$$

$$\sum_{k:a_{sk}=1,a_{tk}=1} x_k = 0, \qquad \text{on the other.} \tag{3.11b}$$

Thus, the branching rule (3.11) forces the rows s and t to be covered by the same column on one branch and by different columns on the other branch. In our problem, this branching scheme has a natural interpretation: on one branch OPOs s and t are forced to be in the same region, and on the other branch, they are forced to be in different regions. Kong calls this *Branching on OPO pairs* [96, 97].

Note that the constraint matrix A of any set-partitioning master problem is a 0-1 matrix. Proposition 3.1 shows that a branching pair can always be identified whenever node solution to the master problem is fractional. Since there are only finitely many pairs of rows, a branch-and-bound algorithm employing this strategy must terminate after a finite number of branches.

Ryan and Foster's branching strategy speeds up the branch-and-bound process by creating a more balanced tree, and eliminating more regions at earlier stages when compared to traditional branching on variables [115]. In standard variable branching, on a branch where a variable is set to 1, a huge number of region designs are eliminated from consideration. On the contrary, on the other branch, where the same variable is set to 0, only a few region designs are eliminated. This results in an unbalanced search tree. However, branching on OPO pairs eliminates roughly equally many potential region designs on both branches, which results in a more balanced branch-and-bound tree.

Furthermore, this branching strategy is well suited for branch-and-price applications. As discussed in [14], one has to be very careful with branching decisions while implementing a branch-and-price algorithm because techniques like traditional variable branching may destroy the structure of the master problem, thus complicating the solution procedure. However, Ryan and Foster's branching strategy can be implemented within the pricing problem, leaving the structure of the master problem untouched.

Note that the full constraint matrices of our master problems, (3.1) and (3.7), are not 0-1 matrices. However, a fractional feasible solution for (3.1) is also a fractional solution for (3.1b), which is a 0-1 matrix. Hence, by Proposition 3.1, a branching pair can always be identified by looking at the rows of the set-partitioning constraints (3.1b). A similar argument also holds for the equity model (3.7). In our implementation, as in [96, 97], during branching after identifying an OPO pair s, t we enforce the branching constraints within the

corresponding pricing problem, i.e., we add $y_s = y_t$ to the pricing problem to force the OPOs to be grouped together on one branch, and add the constraint $y_s + y_t \leq 1$ to the pricing problem on the other branch to force them to be separate. In addition to the changes in the pricing problem, on the *together* branch only columns that satisfy $a_{sk} = a_{tk} = 0$ or $a_{sk} = a_{tk} = 1$ should be active and all other columns should be deleted. Similarly, on the *separate* branch only columns that satisfy $a_{sk} = a_{tk} = 0$ or $a_{sk} = 1, a_{tk} = 0$ or $a_{sk} = 0, a_{tk} = 1$ should be permitted and all others should be deleted.

3.3.2 Approximating the Efficient Frontier

3.3.2.1 An Algorithm Using Parameterization Let η_{min} and λ_{min} denote the efficiency and equity levels of the current regional configuration. We consider only those regional configurations whose efficiency and equity exceed η_{min} and λ_{min}, respectively. Recall that the maximum attainable efficiency value for a given equity measure λ is denoted by $\eta(\lambda)$ and a similar definition holds for $\lambda(\eta)$. Additionally, suppose we denote the objective value of the LP relaxation of formulation (3.1) for λ by $\eta^{LP}(\lambda)$ and the LP relaxation objective value of formulation (3.7) for η by $\lambda^{LP}(\eta)$. Let $\eta_{max} = \eta(\lambda_{min})$, $\eta^{LP}_{max} = \eta^{LP}(\lambda_{min})$, $\lambda_{max} = \lambda(\eta_{min})$ and $\lambda^{LP}_{max} = \lambda^{LP}(\eta_{min})$. Our approach for approximating the actual efficient frontier consists of finding $\eta(\lambda)$, where $\lambda_{min} \leq \lambda \leq \lambda_{max}$ and λ values are guided by sensitivity analysis techniques on the LP relaxations, and enumerating the nondominated solutions on our path. Thus, we are embedding the ε-constraint method within an iterative algorithm to find the Pareto-optimal set of the system.

Suppose that the actual efficient frontier of models (3.1) and (3.7), and the efficient frontier of their LP relaxations are as shown in Figure 3.2(a), where the hollow dots represent Pareto-optimal integer solutions and each of these solutions is actually associated with the whole step that it lies on in the staircase structure. This staircase structure itself is the efficient frontier of the system. The piecewise linear curve that lies atop is the efficient frontier of the LP relaxation of the problem. Each segment of the piecewise linear LP efficient frontier corresponds to an interval of λ where a dual solution of the LP relaxation stays optimal, as discussed in Proposition 3.2.

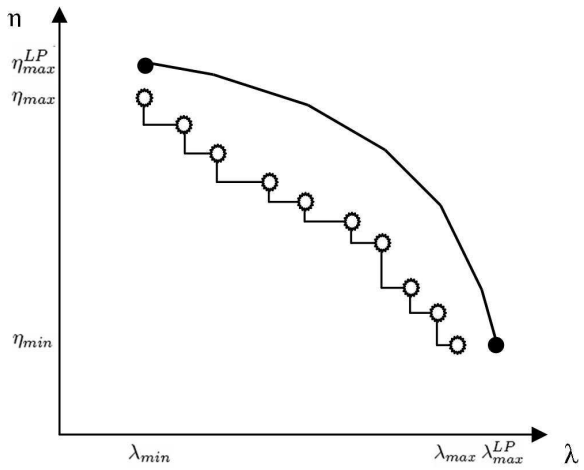


Figure 3.2: A hypothetical representation of the actual efficient frontier for the integer and linear programs.

The procedure that we adopt makes use of upper and lower bounds on the actual frontier that can be calculated using the solutions at hand in any iteration. Obviously, the efficient frontier for the LP is an upper bound on the actual efficient frontier for the IP (as shown in Figure 3.2(a)), but this curve is not known at the beginning of the procedure. Our algorithm partially builds the LP efficient frontier by iteratively creating segments of it and uses this approximation to guide the enumeration of the IP curve. It iteratively constructs the line segments that compose the LP efficient frontier and uses the extended segments to create an upper bound on the actual IP frontier. Propositions 3.2 and 3.3 establish the validity of these upper bounds.

Let $P_{\eta(\lambda)}$ denote the LP relaxation of formulation (3.1) and let $D_{\eta(\lambda)}$ denote the dual of $P_{\eta(\lambda)}$. Recall that π and σ denote the dual variables associated with constraints (3.1b) and (3.1c), respectively. Then $D_{\eta(\lambda)}$ can be written as follows:

$$\min \sum_{i \in I} \pi_i + \lambda \sum_{i \in I} \sigma_i \tag{3.12a}$$

$$\text{subject to} \quad \sum_{i \in I} a_{ir}\pi_i + \sum_{i \in I} f_{ir}\sigma_i \geq c_r, \qquad \forall r \in R', \tag{3.12b}$$

$$\pi_i : \text{free}, \sigma_i \leq 0, \qquad \forall i \in I.$$

Proposition 3.2. *Consider $\lambda_{min} \leq \hat{\lambda} \leq \lambda_{max}$, suppose $P_{\eta(\hat{\lambda})}$ is bounded and feasible, and let $(\hat{\pi}, \hat{\sigma})$ denote the optimal dual solution vector. Then,*

$$\eta^{LP}(\lambda) = \sum_{i \in I} \hat{\pi}_i + \lambda \sum_{i \in I} \hat{\sigma}_i, \qquad \text{for } \lambda \in [\hat{\lambda}_l, \hat{\lambda}_u],$$

where $\hat{\lambda}_l$ and $\hat{\lambda}_u$ denote the lower and upper allowable limits, respectively, provided by sensitivity analysis for the right-hand side parameter λ for the current basis of $P_{\eta(\hat{\lambda})}$ to stay optimal.

Proof. Denote the optimal primal solution for $P_{\eta(\hat{\lambda})}$ by $\hat{x}$, and the associated basis matrix by $\hat{B}$. Note that a change in λ does not affect the reduced costs of primal variables, hence, also leaves the pricing problem unchanged. Consequently, the optimal primal basis and associated dual solutions, $(\hat{B}, \hat{\pi}, \hat{\sigma})$, will stay optimal as long as

$$x_{\hat{B}} = \hat{B}^{-1} \cdot \begin{bmatrix} 1 \\ \vdots \\ 1 \\ \lambda \\ \vdots \\ \lambda \end{bmatrix} \geq 0. \tag{3.13}$$

Hence, by strong duality

$$\eta^{LP}(\lambda) = \sum_{i \in I} \hat{\pi}_i + \lambda \sum_{i \in I} \hat{\sigma}_i, \qquad \text{for } \lambda \in [\hat{\lambda}_l, \hat{\lambda}_u].$$

□

Following Proposition 3.2, we know that for a range of λ where the optimal basis for $P_{\eta(\lambda)}$ remains unchanged, there exists a constant optimal dual solution. Suppose we denote the optimal dual solution associated with the range $[\lambda_l, \lambda_u]$ by $(\pi^{[\lambda_l, \lambda_u]}, \sigma^{[\lambda_l, \lambda_u]})$.

Proposition 3.3. *The extension of a line segment of the LP frontier beyond its allowable range gives a valid inequality for the system in the (η, λ) domain, i.e.,*

$$\sum_{i \in I} \pi_i^{[\hat{\lambda}_l, \hat{\lambda}u]} + \lambda \sum_{i \in I} \sigma_i^{[\hat{\lambda}_l, \hat{\lambda}_u]} \geq \eta^{LP}(\lambda) \geq \eta(\lambda), \qquad \text{for } \lambda_{min} \leq \lambda \leq \hat{\lambda}_l \text{ and } \hat{\lambda}_u \leq \lambda \leq \lambda_{max}.$$

Proof. Suppose $P_{\eta(\lambda)}$ is solved to optimality for a particular $\hat{\lambda}$. Then, from Proposition 3.2, if the problem is bounded and feasible, we have:

$$\eta^{LP}(\lambda) = \sum_{i \in I} \pi_i^{[\hat{\lambda}_l, \hat{\lambda}_u]} + \lambda \sum_{i \in I} \sigma_i^{[\hat{\lambda}_l, \hat{\lambda}_u]}, \qquad \forall \lambda \in [\hat{\lambda}_l, \hat{\lambda}_u]$$

So, $(\pi^{[\hat{\lambda}_l, \hat{\lambda}_u]}, \sigma^{[\hat{\lambda}_l, \hat{\lambda}_u]})$ constitutes a feasible solution for $D_{\eta(\lambda)}$ for any λ since it satisfies (3.12b), and hence, by weak duality, we have:

$$\eta^{LP}(\lambda) \leq \sum_{i \in I} \pi_i^{[\hat{\lambda}_l, \hat{\lambda}_u]} + \lambda \sum_{i \in I} \sigma_i^{[\hat{\lambda}_l, \hat{\lambda}_u]}, \qquad \forall \lambda \notin [\hat{\lambda}_l, \hat{\lambda}_u],$$

which proves that the extension of a segment is an upper bound on the true LP efficient frontier. Since the feasible region of the integer program is smaller than that of its linear relaxation we also know that $\eta(\lambda) \leq \eta^{LP}(\lambda), \forall \lambda$. □

The lower bound on the efficient frontier at any step of the algorithm will basically be constructed by the nondominated integer solutions obtained up to that point. Since the set of generated nondominated integer points represents a subset of the set of all nondominated integer solutions, it constitutes a lower bound for the efficient frontier.

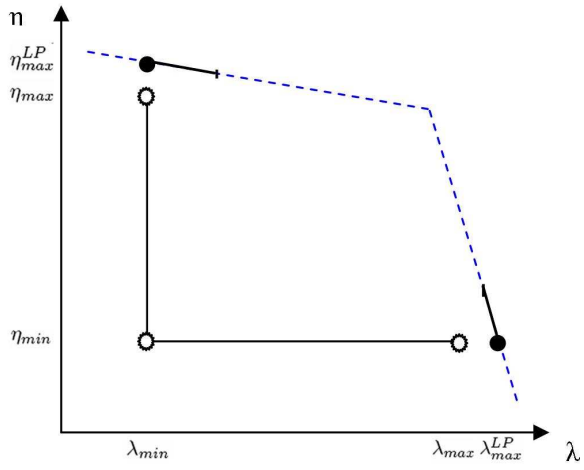
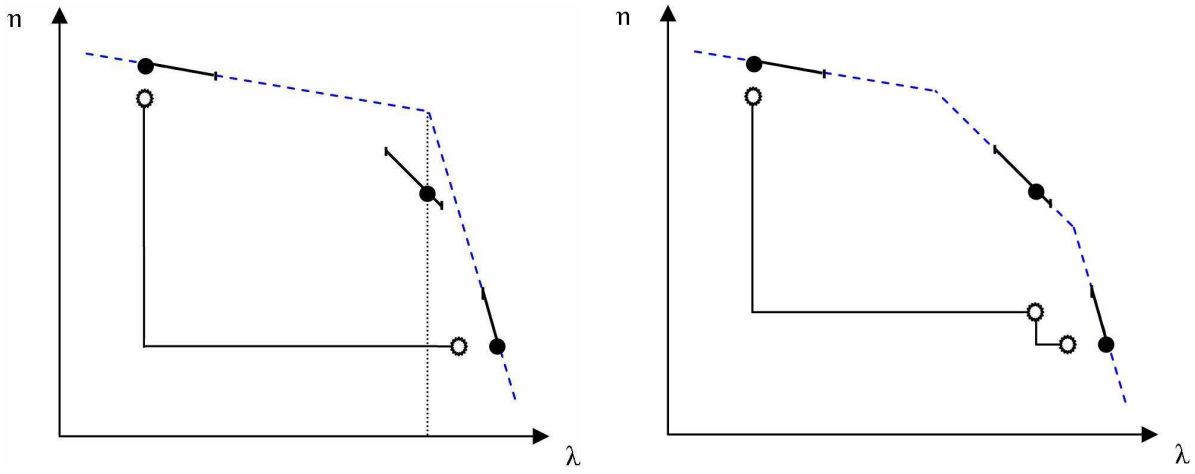


Figure 3.3: An initial "outer envelope" for the efficient frontier based on the LP efficient frontier.

The initialization of our approximation technique is depicted in Figure 3.3(b). The algorithm follows a path in which the LP frontier is constructed step by step while updating

the upper and lower bounds on the IP efficient frontier. In any iteration, after updating the lower and upper bounds, the algorithm proceeds by selecting a λ value where the current upper and lower bounds are farthest apart from each other, constructs a new line segment of the LP frontier around it and updates the bounds appropriately. Thus it selects the λ value where the extensions of the two neighboring line segments that are farthest away from each other intersect. The dual values associated with each segment make it easier to find the intersection points. Figures 3.4 and 3.5 depict two additional iterations of the algorithm and show how the bounds are updated.



(a) (b)

Figure 3.4: Schematic illustration of iterations: (a) One more iteration: a new λ is chosen, the IP and LP are solved, and the new line segment is constructed. (b) Updating the upper and lower bounds.

Before giving a formal description of the algorithm, we introduce more notation. In the description of the algorithm we make use of some attributes associated with a generated line segment. A line segment k, has the following attributes that should be calculated when it's generated:

- λ'_k: the lower endpoint of the segment,
- λ''_k: the higher endpoint of the segment,
- (π^k, σ^k): the dual vector associated with the line segment (recall that each line segment in the efficient frontier of the LP corresponds to one dual solution by Proposition 3.2).
- For each integer solution found on the path the following information should be stored:

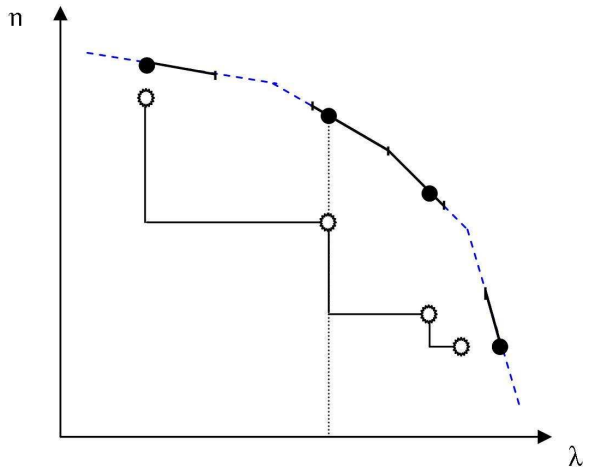


Figure 3.5: After one more iteration.

– λ_k: the point in the segment for which an integer solution has been found,

– x^k: the primal optimal solution of the efficiency model for λ_k.

Furthermore we construct two sets that sort the intervals and the values of λ for which integer solutions have been found and store them in order. Namely, the *Ordered Segment Set*, $\mathcal{S}$, and *Ordered Integer Solution Point Set*, $\mathcal{P}$, store the pairs of endpoints for each constructed segment and the λ values for which an integer solution has been computed in an increasing order, respectively. At each iteration, after the inclusion of new members, both of these sets will be reordered and reindexed. The notations $(\lambda'_{(i)}, \lambda''_{(i)})$ and $\lambda_{(i)}$ will be used denote the i^{th} members of $\mathcal{S}$ and $\mathcal{P}$ at any point.

Finally, we define a set of tolerance parameters to be used throughout the algorithm. Let ϵ_L, ϵ_η and ϵ_λ denote the tolerance parameters for the LP efficient frontier, efficiency objective value comparison and enumerated integer point distance, respectively.

Now we are ready to list the steps of the algorithm:

Algorithm 3.1. *Approximating the Efficient Frontier*

1. Initialization.

a. Find η_{min} and λ_{min} from the currently used system.

b. Calculate $\eta_{max} = \eta(\lambda_{min})$ and $\eta^{LP}_{max} = \eta^{LP}(\lambda_{min})$.

Construct Segment 1*, namely S_1:*

- *Set* $\lambda_1' = \lambda_{min}$,
- *Set* λ_1'' *to the largest allowable* λ *value so that the current LP basis stays optimal,*
- *Set* $\lambda_1 = \lambda_{min}$,

Let $\mathcal{S} = \{S_1\}$ *and* $\mathcal{P} = \{\lambda_1\}$.

c. *Calculate* $\lambda_{max} = \lambda(\eta_{min})$ *and* $\lambda_{max}^{LP} = \lambda^{LP}(\eta_{min})$.

Construct S_2*:*

- *Set* λ_2' *to the lowest allowable* λ *value so that the current LP basis stays optimal,*
- *Set* $\lambda_2'' = \lambda_{max}^{LP}$,
- *Set* $\lambda_2 = \lambda_{max}$.

Put S_2 *into* $\mathcal{S}$*. Reorder the set.*

Put λ_2 *into* $\mathcal{P}$*. Reorder the set.*

d. *Set the segment counter* $k = 3$ *and go to Step 2*

2. Choosing a new λ. *Choose two sequential members of* $\mathcal{S}$ *that are farthest away from each other.*

a. *Choose*

$$l^* \in \arg\max_{(l):S_{(l)}\in\mathcal{S}, l<|\mathcal{S}|}(\lambda'_{(l+1)} - \lambda''_{(l)}) \tag{3.14}$$

b. *If*

$$\lambda'_{(l^*+1)} - \lambda''_{(l^*)} < \epsilon_L, \tag{3.15}$$

the enumeration of the LP efficient frontier is complete; go to Step 4.

Otherwise, go to Step 3.

3. Constructing S_k.

a. *Calculate*

$$\lambda_k = \frac{(\sum_{i\in I}\pi_i^{(l^*+1)}) - (\sum_{i\in I}\pi_i^{(l^*)})}{(\sum_{i\in I}\sigma_i^{(l^*)}) - (\sum_{i\in I}\sigma_i^{(l^*+1)})} \tag{3.16}$$

b. *Find* $\eta(\lambda_k)$ *and* $\eta^{LP}(\lambda_k)$,

c. *Set* λ_k' *to the lowest allowable* λ *value so that the current LP basis stays optimal,*

d. *Set* λ_k'' *to the largest allowable* λ *value so that the current LP basis stays optimal.*

Put S_k *into* $\mathcal{S}$*. Reorder the set.*

Put λ_k *into* $\mathcal{P}$*. Reorder the set.*

Set $k = k + 1$.

Go to Step 2.

4. Generating more integer points

   a. *Choose two sequential members of* $\mathcal{P}$ *that are farthest apart from each other.*

      i. *Choose*

$$l^* \in \arg\max_{(l):\lambda_{(l)}\in\mathcal{P}, l<|\mathcal{P}|}\{\lambda_{(l+1)} - \lambda_{(l)} | \eta(\lambda_{(l)}) - \eta(\lambda_{(l+1)}) < \epsilon_\eta\} \tag{3.17}$$

      ii. *If*

$$\lambda_{(l^*+1)} - \lambda_{(l^*)} < \epsilon_\lambda, \tag{3.18}$$

      the approximation of the IP efficient frontier is complete; STOP.

      The approximate frontier is given by the following set:

$$\eta = \begin{cases} \eta^*(\lambda_{(i+1)}), & \text{if } \lambda_{(i)} < \lambda \leq \lambda_{(i+1)}, i \leq |\mathcal{P}| - 1, \\ \eta^*(\lambda_{(1)}), & \text{if } \lambda = \lambda_{(1)}, \\ 0, & \text{otherwise} \end{cases} \tag{3.19}$$

      Otherwise go to Step 4b.

   b. *Calculate*

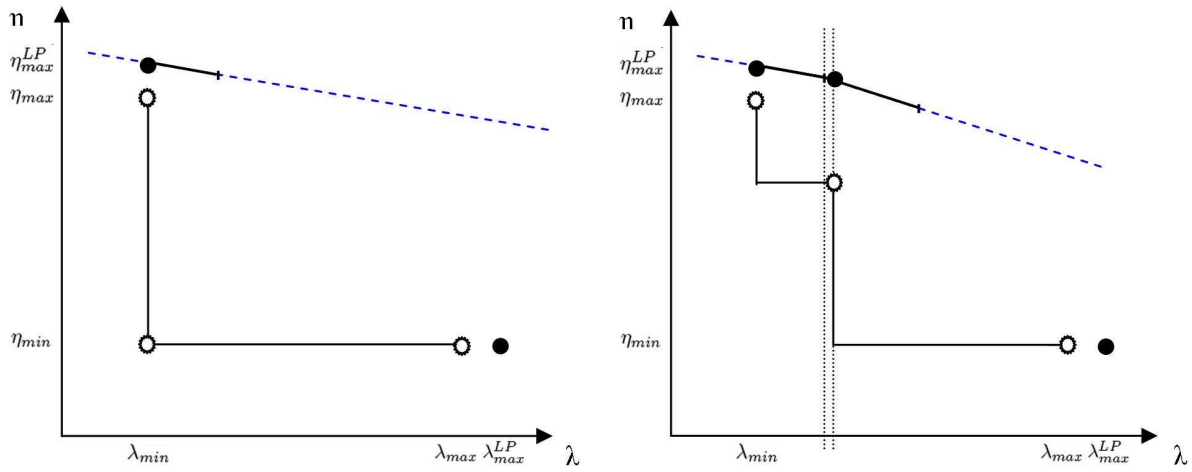
$$\lambda_k = \frac{\lambda_{(l^*)} + \lambda_{(l^*+1)}}{2} \tag{3.20}$$

      Find $\eta(\lambda_k)$ *and* $\eta^{LP}(\lambda_k)$.

      Put λ_k *into* $\mathcal{P}$. *Reorder the set.*

      Set $k = k + 1$.

      Go to Step 4a.



(a) (b)

Figure 3.6: (a) Initialization of the alternative algorithm. (b) An additional iteration of the alternative algorithm: a new λ is chosen, the new line segment is constructed, and bounds are updated accordingly.

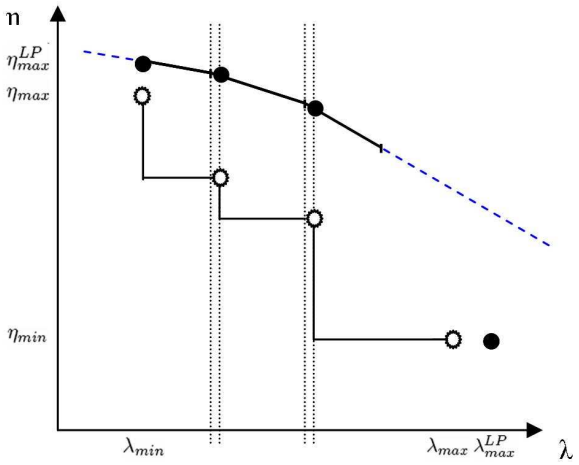


Figure 3.7: After one more iteration of the alternative algorithm.

3.3.2.2 An Alternative Algorithm Another possible way to trace the efficient frontier of the LP and enumerate integer solutions on the way is to construct the curve by finding the line segments that form the piecewise linear LP frontier in a sequential manner. In this algorithm, we start by forming the first line segment around λ_{min} and whenever a segment k is formed, we construct the next segment around $\lambda_{k+1} = \lambda_k'' + \epsilon$, if $\lambda_k'' + \epsilon \leq \lambda_{max}$. Thus, we construct the line segments forming the efficient frontier of the LP sequentially by perturbing

the higher endpoints of each line segment that we find on our path. Once again, we solve the IP and enumerate the integer solutions for each λ_k for all $S_k \in \mathcal{S}$. Figures 3.6 and 3.7 show schematic representations of the iterations of the alternative algorithm.

Algorithm 3.2. *An Alternative Algorithm for Approximating the Efficient Frontier*

1. Initialization.

   a. *Find* η_{min} *and* λ_{min} *from the currently used system.*

   b. *Calculate* $\eta_{max} = \eta(\lambda_{min})$ *and* $\eta^{LP}_{max} = \eta^{LP}(\lambda_{min})$.
   Construct Segment 1, *namely* S_1*:*

   - *Set* $\lambda'_1 = \lambda_{min}$,
   - *Set* λ''_1 *to the largest allowable* λ *value so that the current LP basis stays optimal,*
   - *Set* $\lambda_1 = \lambda_{min}$,

   Let $\mathcal{P} = \{\lambda_1\}$.

   c. *Calculate* $\lambda_{max} = \lambda(\eta_{min})$ *and* $\lambda^{LP}_{max} = \lambda^{LP}(\eta_{min})$.
   Put λ_{max} *into* $\mathcal{P}$.

   d. *Set the segment counter* $k = 2$ *and go to Step 2*

2. Choosing a new λ. *Set* $\lambda_k = \lambda''_{k-1} + \epsilon$. *If* $\lambda_k \geq \lambda_{max}$ *the enumeration of the LP efficient frontier is complete; go to Step 4. Otherwise, go to Step 3.*

3. Constructing S_k.

   - *Find* $\eta(\lambda_k)$ *and* $\eta^{LP}(\lambda_k)$,
   - *Set* $\lambda'_k = \lambda''_{k-1}$,
   - *Set* λ''_k *to the largest allowable* λ *value so that the current LP basis stays optimal.*

   Put λ_k *into* $\mathcal{P}$. *Reorder the set.*
   Set $k = k + 1$.
   Go to Step 2.

4. Generating more integer points
   Follow Step 4 of Algorithm 3.1.

3.4 COMPUTATIONAL RESULTS

The optimization models were coded in C++ using COIN/BCP, an open source branch, cut and price framework which is part of the COIN-OR, i.e., Computational Infrastructure for Operations Research, project [37]. For a comprehensive description of COIN/BCP, the reader is referred to [38, 140, 141]. In the application developed, COIN/BCP stays in charge of the branch-and-bound tree management while it uses CPLEX 9.0 to solve the subproblems throughout the execution of the algorithm. We used a UNIX machine with AMD Opteron 240 processor and 3.8 GB RAM.

3.4.1 Data Sources and Parameter Estimation

The main sources of data were the UNOS website and the UNOS Data Set available at University of Pittsburh Medical Center that covered a time frame from the year 1988 to the end of 2002 [169]. The discrete-event simulation model for End-Stage Liver Disease and organ allocation of Shechter et al. [155] was modified to capture information on pure distribution likelihood and the pure national flow likelihood estimates.

The national liver allocation system was simulated between the years 1999 and 2002 with no regions specified, i.e., one single national waiting list that ranks patients according to severity of illness. In this case, l_{ij} was estimated as the percentage of organs donated at OPO i and transplanted to a patient in OPO j averaged over 100 independent replications. The pure national flow likelihood parameter for OPO i, i.e., l_i^0, was estimated using 50 randomly selected region configurations, with number of regions between 5 and 14, and replicating the simulation 30 times for each configuration.

Table 3.1: Summary of geographic decomposition schemes used.

Geographic Decomposition Scheme ID	Number of		Pricing Problem Analysis		
	Region Covers	OPOs in Each Cover	Variables		Constraints
			Binary	Total	
10_10	10	10	10	155	1055
20_10	20	10	10	155	1055
20_11	20	11	11	187	1397
20_12	20	12	12	222	1806
20_15	20	15	15	345	3495

For the measure of organ quality decay with respect to CIT we used the model introduced by Totsuka et al. [164], which was also used in previous studies in liver transplantation region design [96, 97, 157]. Thus, we consider PNF, i.e., *primary non-function* as the main source of postoperative organ failure, and use the following formulas to measure the liver quality decay with respect to CIT (cold-ischemia time) and OTD (organ transport distance):

$$CIT = 9.895 + 0.003 \times OTD, \tag{3.21a}$$

$$PNF = -1.5545 + 1.17799 \times CIT - 0.03451 \times CIT^2 + 0.0004 \times CIT^3. \tag{3.21b}$$

Let $OTD(i, j)$ denote the organ transport distance in miles between donor OPO i and recipient OPO j. Then, the viability-adjustment multiplier between OPOs i and j, α_{ij}, is calculated as

$$\alpha_{ij} = 1 - PNF(CIT(OTD(i, j)). \tag{3.22}$$

3.4.2 Efficient Frontier Approximations

We executed the algorithms discussed in Sections 3.3.2.1 and 3.3.2.2 using different geographic decomposition schemes with different region covers. Table 3.1 summarizes details on the geographic decomposition schemes that have been utilized to solve the efficiency and

Table 3.2: CPU times for different geographic decomposition schemes.

Geographic Decomposition Scheme ID	Average CPU Time to Get a Step of the Frontier (sec)	Total CPU Time to Enumerate the Frontier (hour:min:sec)
10_10	546	00:27:18
20_10	1080	01:12:00
20_11	4254	04:43:36
20_12	10972	12:11:28
20_15	22809	38:00:54

equity models. The region covers are named in such a way that region cover K_L contains K geographic subsets, each containing L OPOs, following the notation in [96, 97]. This table also shows data related to the pricing problems associated with each region cover in each decomposition scheme. Recall that every region cover has its own pricing problem, and thus, a scheme that employs K region covers will have K pricing problems. The numbers displayed are for the pricing problem of the efficiency model (3.6), which has L more constraints than the pricing model of the equity model (3.9), everything else being equal.

We came up with these sets of region covers after doing considerable initial testing on different schemes. We used Kong's geographic decomposition schemes [96] as a starting point and finalized our sets of region covers after gradually modifying them based on the results from initial testing.

During the execution of the enumeration algorithms we stored all column generated during one solution of an IP for some value of λ and inserted those into the initial set columns for subsequent solutions with new λ values. Initially, we started solving the first IP with a set of columns that includes the current regional configuration, all single-OPO regions, and all contiguous regions with 4 OPOs.

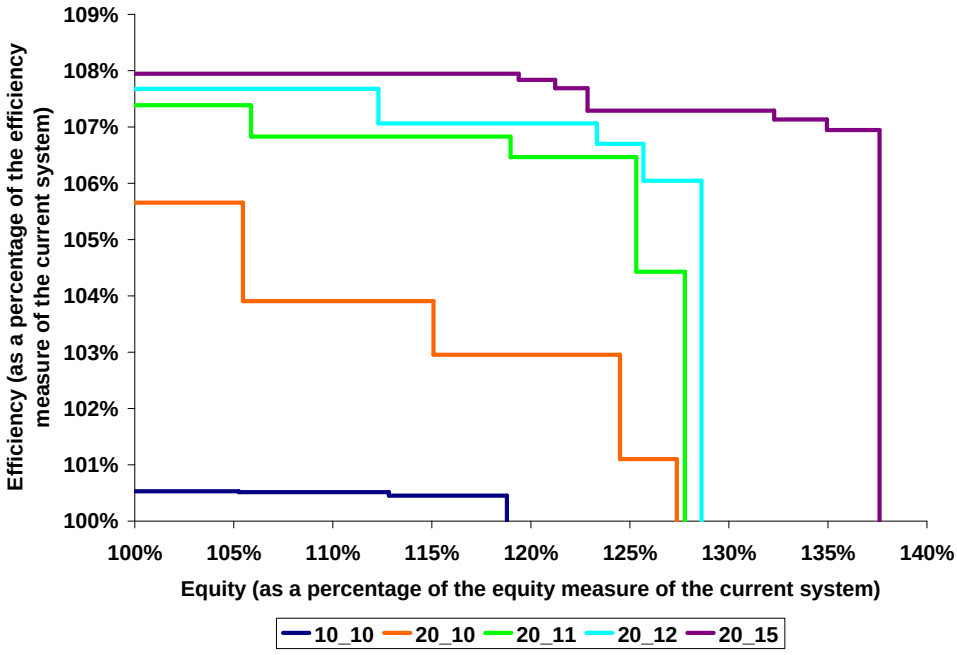


Figure 3.8: A summary of efficient frontiers obtained by analyzing the system using different region covers for the geographic decomposition scheme.

The solution times obtained with our geographic decomposition schemes and the associated efficient frontiers can be seen in Table 3.2 and Figure 3.8, respectively. For every scheme, Table 3.2 shows both the average CPU time in seconds to solve a parametric IP in order to enumerate a step of the efficient frontier, and the total CPU time spent on constructing the full frontier. As can be seen from Figure 3.8, the results we obtained using geographic decomposition scheme 20_15 are the best. However, enumerating the efficient frontier using 20_15 is computationally expensive, requiring about 38 hours of CPU time. We decided not to design bigger region covers after observing the jump in CPU time from scheme 20_12 to 20_15.

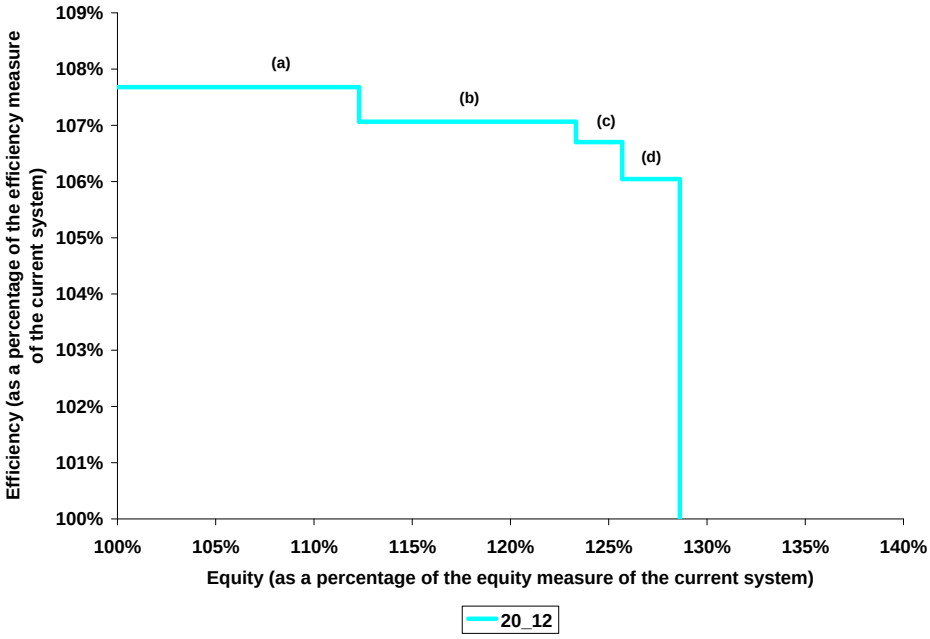
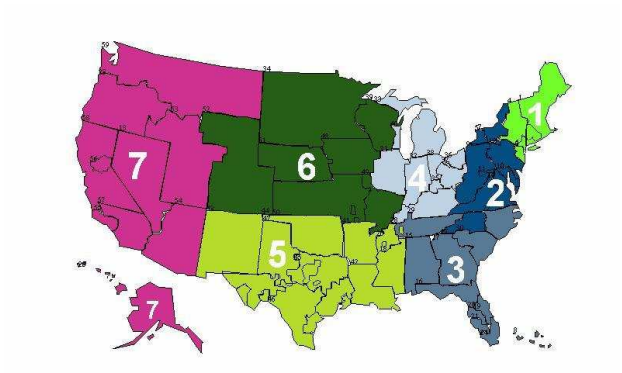
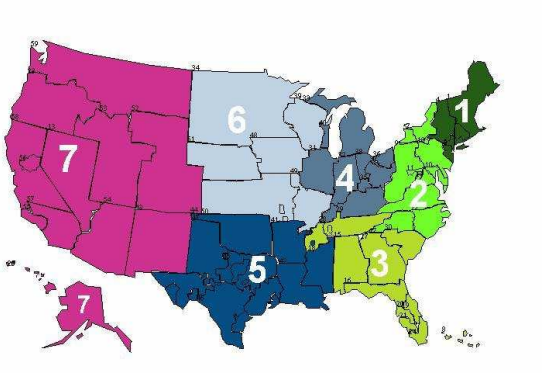


Figure 3.9: Efficient frontier obtained with geographic decomposition scheme 20_12.

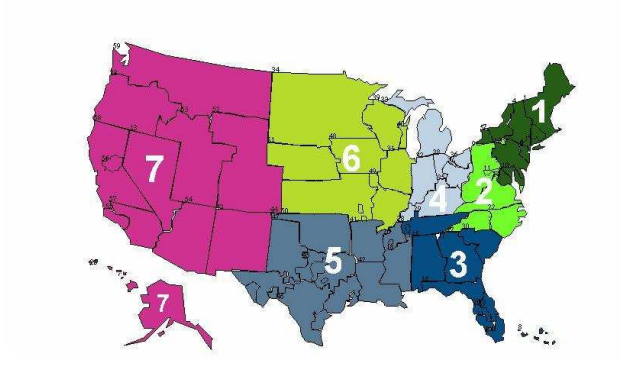
We present the efficient frontier obtained with geographic decomposition scheme 20_12 and the corresponding regional configurations in Figures 3.9 and 3.10, respectively. Similarly, the efficient frontier obtained using geographic decomposition scheme 20_15 and the corresponding regional configurations can be found in Figures 3.11 and 3.12, respectively. Figure 3.11 also compares our set of Pareto-optimal solutions to alternative regional configurations provided by Kong et al. [97] and Stahl et al. [157]. In both studies, a number of results were mentioned without listing any details regarding what the individual groupings of OPOs were. However, both papers included a map of recommended regional configurations, which we evaluated and compared with our own results.



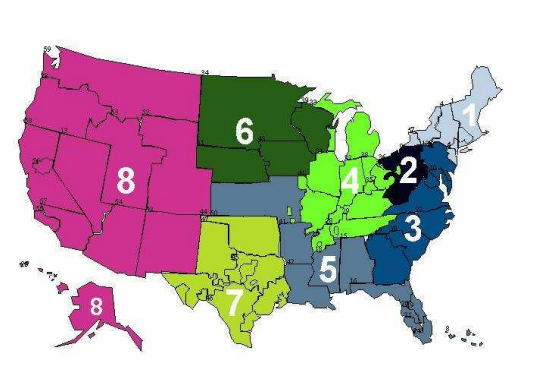
(a) Efficiency: 7.68% increase
Equity: 12.3% increase



(b) Efficiency: 7.06% increase
Equity: 23.33% increase



(c) Efficiency: 6.70% increase
Equity: 25.69% increase



(d) Efficiency: 6.04% increase
Equity: 28.62% increase

Figure 3.10: Maps of regions that correspond to the steps of the efficient frontier obtained with geographic decomposition scheme 20_12.

3.4.3 Evaluating the Regional Configurations

In order to validate our solutions, we simulated our alternative configurations of Figure 3.12 and compared the results to the performance of the current regional configuration. We used the simulation model of Shechter et al. [155], and simulated the liver allocation system from 1996 to the end of 2002. We took 20 replications using each configuration and used a warm-up period of 3 years. To demonstrate the significance of gains with respect to both objectives, i.e., maximizing efficiency and maximizing minimum equity, we ran paired t tests to compare the performances of our regional configurations with the current system. We also simulated the regional configurations provided by Stahl et al. [157] and Kong et al. [97].

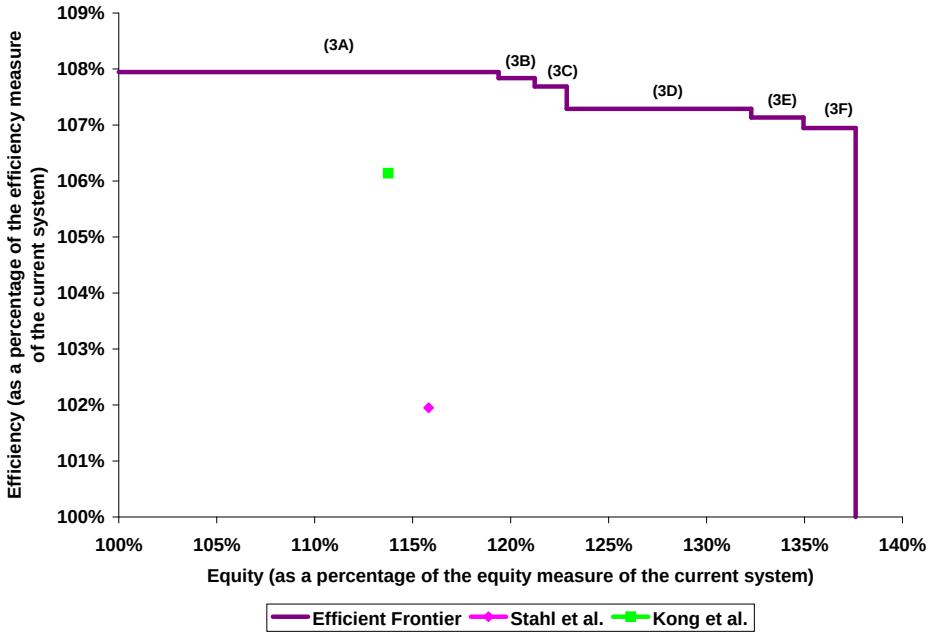


Figure 3.11: Efficient frontier obtained with geographic decomposition scheme 20_15. Our results are also compared to solutions provided by Stahl et al. [157] and Kong et al. [97].

Table 3.3 shows the results of paired t tests for the difference in mean number of transplants. Each row in the table corresponds to a paired t test comparing the performance of an optimal regional configuration in Figure 3.12 to the current system. The null hypothesis of these tests is that the average number of transplants are equal between an alternative configuration and the current regional configuration. Since the p value is 0 for each configuration, we conclude that there is enough statistical evidence that shows that the configurations shown in Figure 3.12 result in an increase in the average number of yearly transplants. Figures 3.13 and 3.14 depict the 95% confidence intervals shown in Table 3.3, and also display the confidence intervals obtained for the solutions presented by Stahl et al. [157] and Kong

et al. [97]. As can be seen from these values, the regional configurations presented in Figure 3.12 result in a 5% increase in the number of yearly transplants on the average while the alternative configurations of Stahl et al. [157] and Kong et al. [97] result in an average of about 3% and 4.5% increase, respectively.

Table 3.4 displays the results of paired t tests for the difference in mean minimum intra-regional transplant rates per patient. As in Table 3.3, each row in the table corresponds to a paired t test comparing the performance of an optimal regional configuration in Figure 3.12 to the current system. Once again, since the p value is 0 for each configuration, we conclude that there is enough statistical evidence that shows our regional configurations result in an increase in the minimum rate of intra-regional transplants per patient. Figures 3.15 and 3.16 display the 95% confidence intervals shown in Table 3.4 along with the intervals constructed for the solutions provided by Stahl et al. [157] and Kong et al. [97]. Although the amount of increase in the minimum equity level looks too small, the average minimum equity level for the current system is around 0.017 and our alternative region designs result in a 70% increase. As can be seen from Figures 3.15 and 3.16, the solution presented in Stahl et al. [157] results in a 40% increase in the equity measure. The paired t test for the regional configuration presented in Kong et al. [97] has a p value of 0.114, which suggests that there is not enough statistical evidence that this configuration will improve the current equity levels.

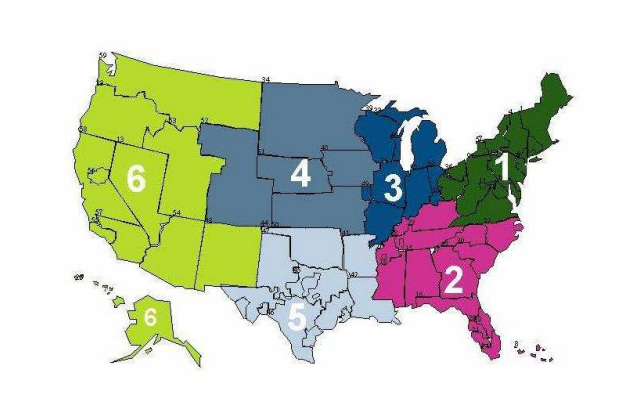
3.5 CONCLUSIONS

In this chapter, we extended the models and solution techniques used for the region design problem in the liver allocation hierarchy in the healthcare optimization literature. We introduced an optimization framework that aimed to balance the efficiency and equity in the system through the use of two parametric integer programs.

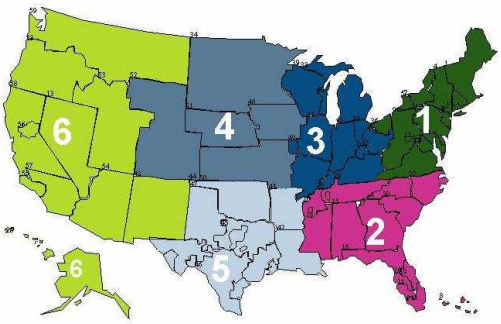
Our results indicate that there are alternative regional configurations that would benefit the society by bringing gains to both the efficiency, as measured by the number of viability-adjusted transplants, and equity, as measured by the fairness in terms of access to organs from

different geographic locations throughout the country, of the national liver allocation system. The regional configurations presented in this chapter are estimated to bring 5% and 70% gains in the efficiency and equity levels of the U.S. liver allocation system, respectively, when tested using the simulation model of Shechter et al. [155]. Furthermore, their performances appear to be superior when compared to alternative configurations suggested in the literature [97, 157].

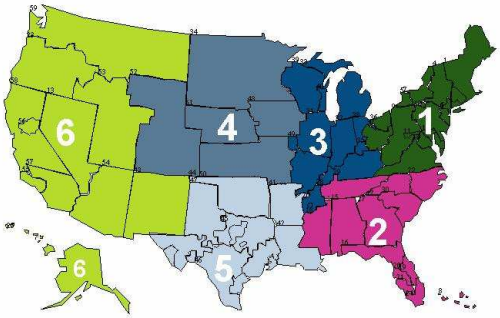
Possible directions of future research are relaxing the steady-state assumption in the modeling phase and enabling a stochastic view of the system that explicitly accounts for uncertainty. Moreover, in this study we have focused on geographic equity in a maxi-min setting. Although alternative equity measures, such as racial, socioeconomical equity, etc. are possible, no attempt has been done to integrate these factors into a decision-making perspective for designing regional configurations for the liver allocation hierarchy.



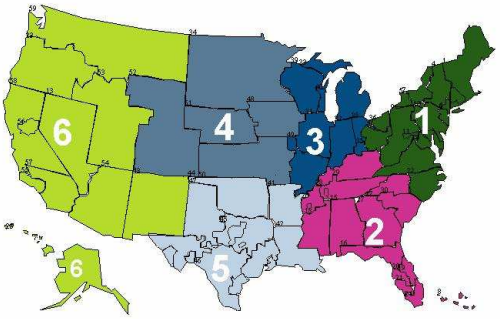
(3A) Efficiency: 7.95% increase
Equity: 19.35% increase



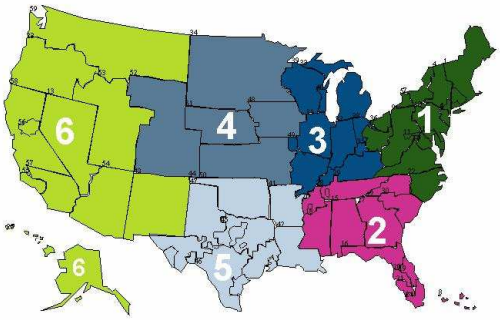
(3B) Efficiency: 7.84% increase
Equity: 21.06% increase



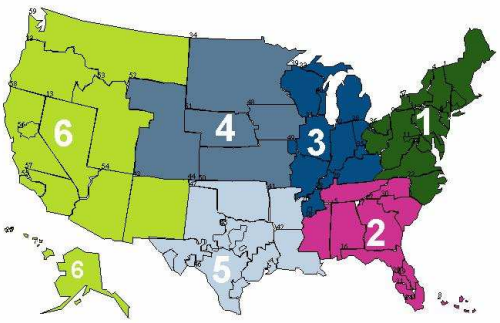
(3C) Efficiency: 7.69% increase
Equity: 22.82% increase



(3D) Efficiency: 7.29% increase
Equity: 32.17% increase



(3E) Efficiency: 7.13% increase
Equity: 34.84% increase



(3F) Efficiency: 6.95% increase
Equity: 37.6% increase

Figure 3.12: Maps of regions that correspond to the steps of the efficient frontier obtained with geographic decomposition scheme 20_15.

Table 3.3: Paired t tests and 95% confidence intervals on the difference in average number of transplants per year: Regional configurations of Figure 3.12 vs. current system

Regional Configuration	Paired Differences					t	$p-$value
	Mean	Standard Deviation	Standard Error	95% CI LL	95% CI UL		
3A	203.53	29.26	6.54	189.84	217.23	31.11	.0000
3B	204.95	31.91	7.14	190.01	219.89	28.72	.0000
3C	209.49	41.07	9.18	190.27	228.71	22.81	.0000
3D	197.32	38.92	8.70	179.11	215.54	22.67	.0000
3E	178.71	39.76	8.89	160.10	197.33	20.10	.0000
3F	196.08	35.29	7.89	179.57	212.60	24.85	.0000

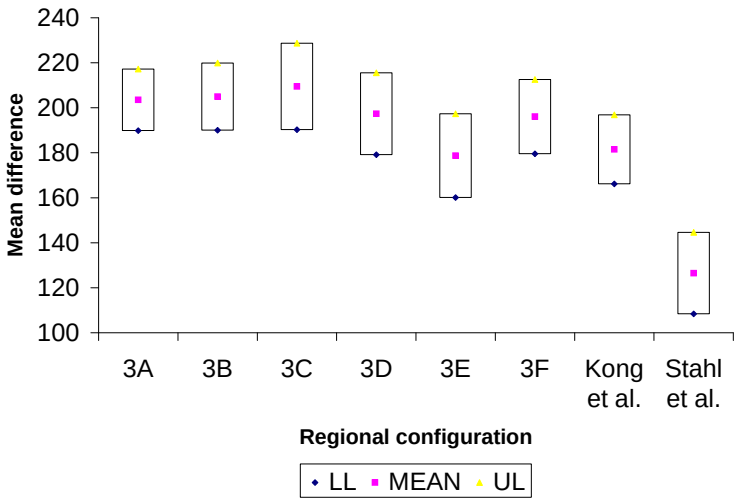


Figure 3.13: 95% confidence intervals around the mean difference in number of transplants. Solutions provided by Stahl et al. [157] and Kong et al. [97] are also evaluated. Detailed results can be seen in Table 3.3.

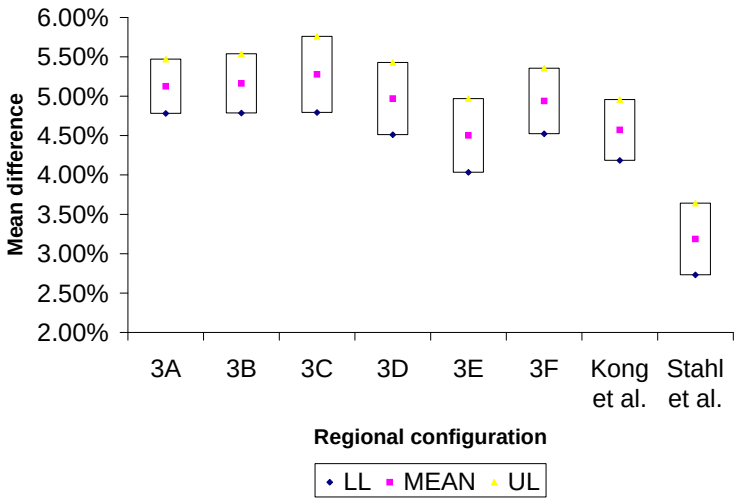


Figure 3.14: 95% confidence intervals around the mean difference in number of transplants as a percentage of transplants under the current system. Detailed results can be seen in Table 3.3.

Table 3.4: Paired t tests and 95% confidence intervals on the difference in average minimum intra-regional transplant rate per patient per year: Regional configurations of Figure 3.12 vs. current system

Regional Configuration	Paired Differences					t	$p-$value
	Mean	Standard Deviation	Standard Error	95% CI LL	95% CI UL		
3A	.0126	.0062	.0014	.0097	.01553	9.0336	.0000
3B	.0120	.0039	.0009	.0102	.01385	13.8707	.0000
3C	.0127	.0056	.0013	.0100	.01530	10.0699	.0000
3D	.0129	.0044	.0010	.0108	.01497	12.9718	.0000
3E	.0132	.0055	.0012	.0106	.01575	10.7129	.0000
3F	.0108	.0040	.0009	.0089	.01262	12.0087	.0000

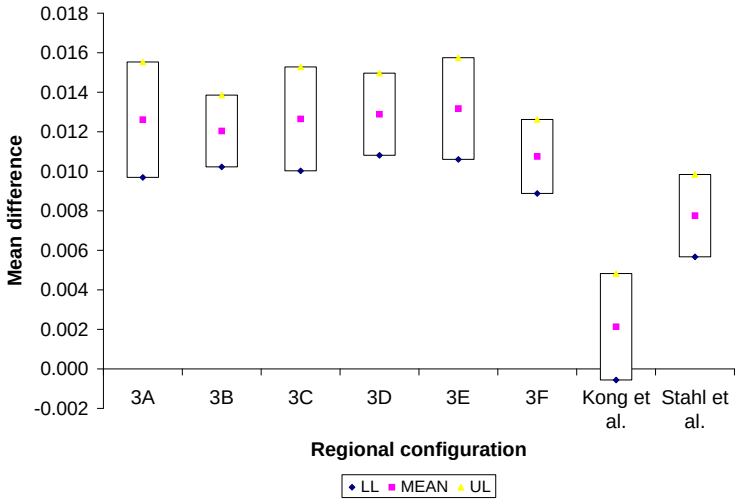


Figure 3.15: 95% confidence intervals around the mean difference in minimum intra-regional transplant rate per patient. Detailed results can be seen in Table 3.4.

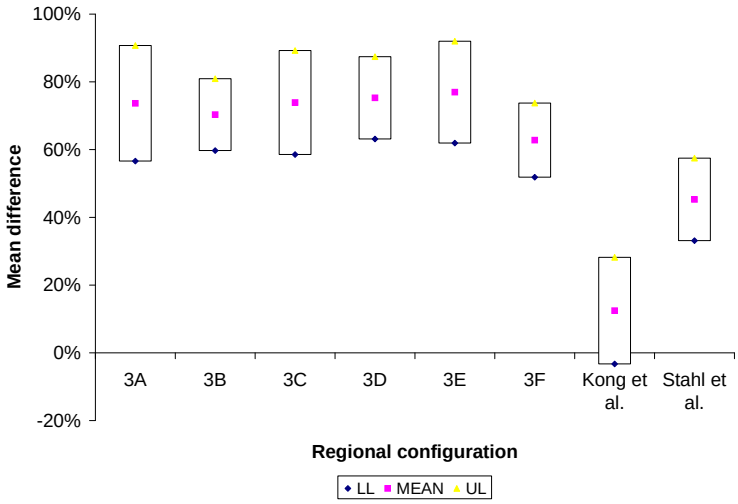


Figure 3.16: 95% confidence intervals around the mean difference in minimum intra-regional transplant rate per patient as a percentage of the equity measure under the current system. Detailed results can be seen in Table 3.4.

4.0 DESIGNING LIVER TRANSPLANT REGIONS UNDER UNCERTAINTY USING A PATIENT-BASED MODEL

4.1 INTRODUCTION

Previous studies on the design of regions in the U.S. for the liver transplantation mechanism [96, 97, 157] modeled the current system under steady-state assumptions, assuming constant flow of organs between OPO pairs, organ donations and patient populations. In this chapter, we lift the steady-state assumption and adopt a stochastic view of the system.

We start by constructing a two-stage stochastic recourse model, where the first-stage decision is to design a regional configuration, and the second stage approximates the expected benefit of a liver transplant under different scenarios that represent potential snapshots of the waiting list. We show that a closed-form expression is available for the second-stage expected value function, and concentrate on the resulting set-partitioning formulation. We utilize a column generation approach on the set-partitioning formulation to find favorable region designs. We formulate a pricing problem that explicitly considers a set of possible scenarios for the composition of the national waiting list and generates promising regions.

We also adopt a different objective than all other previous studies on the region design problem [96, 97, 157] for this chapter. We turn our attention from the number of (viability-adjusted) transplants per year to the expected life-time gained from transplants. Hence, the definition of *efficiency* throughout this chapter is different than our definition in Chapter 3, and focuses on the expected life-days gained after the patients receive transplants.

In this chapter, as in Chapter 3, we focus our attention to MELD patients on the waiting list and ignore Status 1 patients. And similarly, we exclude the national matching phase from consideration. See Section 1.2 for more information.

The chapter is structured as follows: in Section 4.2, we introduce a stochastic programming model to maximize the efficiency of the regional configuration of OPOs for liver transplantation in the U.S. We also discuss our column generation approach and introduce our pricing problem formulation. We focus on the solution methods for the resulting models and discuss various computational approaches in Section 4.3. Section 4.4 describes the details about data sources and parameter estimation. We present our computational results in Section 4.5. Finally, in Section 4.6 we summarize our discussions and results.

4.2 A STOCHASTIC PROGRAMMING MODEL FOR REGION DESIGN UNDER UNCERTAINTY

Let I denote the set of OPOs across the U.S., R be the set of all potential regions of OPOs and I_r denote the set of OPOs in region $r, \forall r \in R$. Suppose Ξ denotes the finite support of all scenarios, which represent potential states of the national waiting list. We assume that the scenario space can be represented by a set of discrete scenarios, denoted by $\xi^1, \ldots, \xi^K$, where scenario ξ^k occurs with probability p^k.

There are some important points that should be mentioned regarding the scenarios. First, scenarios give different snapshots of the waiting list, and show possible compositions of the national waiting list at discrete points in time. Suppose that in each scenario there is one harvested liver at some OPO that is about to be matched with patients on the waiting list at the regional level. Each scenario defines the characteristics of the harvested liver and the characteristics of the patients on the waiting list (i.e., their physical and demographic characteristics, which come into play in the ranking of patients and the expected benefit resulting from giving the liver to a specific patient). Regarding the probability distribution for scenarios, we state the following assumption for computational tractability:

Assumption 4.1. *The probability distribution of scenarios is independent of the regional configuration, i.e., p^k is independent of the first-stage decision x, for $\xi^1, \ldots, \xi^K$.*

A detailed discussion on the structures of scenarios will be given during the discussion of the second-stage model.

4.2.1 First-Stage Model

Let $x_r = 1$ if region r is selected for the regional configuration, and $x_r = 0$ otherwise. Also let $a_{ir} = 1$ if OPO i is in region r, and $a_{ir} = 0$ if not.

Then, the first-stage problem can be formulated as follows:

$$\max \mathbb{E}_\xi Q(x, \xi^k) \tag{4.1a}$$

$$\text{subject to} \sum_{r \in R} a_{ir} x_r = 1, \qquad \forall i \in I, \tag{4.1b}$$

$$x_r \in \{0, 1\}, \qquad \forall r \in R, \tag{4.1c}$$

where $Q(x, \xi^k)$ denotes the expected outcome of liver transplantation for regional configuration x under scenario ξ^k. The objective function (4.1a) is composed only of the expectation of the second-stage objective over all possible scenarios. There is no benefit for a particular region enjoyed in the first stage. The objective is to maximize the expected benefit of liver transplantation, which will be clarified during the discussion on the second-stage models. The set-partitioning constraint (4.1b) ensures that each OPO is contained within one single region in the regional configuration.

4.2.2 Second-Stage Model

For $k = 1, \ldots, K$, scenario ξ^k lists an organ ready to be matched at the regional phase, and a list of patients on the waiting list along with information on their OPOs and the expected benefit each patient will get if she is offered the liver.

Let $M_k = \{1, \ldots, |M_k|\}$ denote the set of patients on the waiting list under scenario ξ^k. We assume that patients are numbered so that if $m' < m''$, patient m' is offered the liver before patient m''. This means that patient m' is either sicker than m'', i.e., has a higher MELD score, or their MELD scores are equal but m' has higher UNOS points due to a better blood-type match, longer waiting time on the list, etc. See Section 1.1 for details on

the liver allocation system. Let $O(k) \in I$ denote the OPO in which the liver is harvested under scenario ξ^k. For every $m \in M_k$, let $O(m,k) \in I$ denote the OPO where patient m in scenario ξ^k is listed. Since we assume that the donated liver in OPO $O(k)$ is available for regional matching, i.e., was not matched locally within OPO $O(k)$, we do not need to consider patients listed in OPO $O(k)$ for scenario ξ^k. Thus, $\{m \in M_k | O(m,k) = O(k)\} = \emptyset$.

Let $c(\xi^k) \in \mathbb{R}_+^{|M_k|}$ be the *expected benefit* vector where each entry, $c_m(\xi^k)$, is the expected benefit to patient m given that the liver is transplanted to patient m [144]. Let $T(\xi^k) \in \mathbb{R}^{|M_k| \times |R|}$ be the *probability-ranking matrix* where $T_{mr}(\xi^k) = Prob\{m|r,k\} = Prob\{$patient m gets the liver|region r chosen under scenario $\xi^k\}$. In every scenario ξ^k, the characteristics of the harvested liver and patients are reflected in the problem through the expected benefit vector $c(\xi^k)$ and the probability-ranking matrix $T(\xi^k)$.

We also utilize the following assumption about patient sets:

Assumption 4.2. *Patients do not multiply list, i.e., a patient registers for the national liver waiting list only in one OPO.*

Under the current policy, patients are actually allowed to multiply list, and around 3.3% of ESLD patients in the U.S. do so [119]. However, Assumption 4.2 is not a limiting assumption since a multiply-listed patient can simply be treated as two different patients that share the same characteristics.

Modeling the Liver Allocation Probability Distribution

Before getting into the calculation of $Prob\{m|r,k\}$ we introduce more notation.

Let $J_k \in \mathbb{Z}_+$ denote the maximum number of patients that we can offer the liver at the regional phase due to CIT limitations under scenario ξ^k. Let q_k denote the probability that a patient accepts an organ offer under scenario ξ^k.

We state the following assumption, which was also utilized in Shechter et al. [155], regarding transplant probabilities, which we will revisit shortly:

Assumption 4.3. *For every scenario $\xi^k \in \Xi$, the rank of the patient that will receive the transplantation has a geometric distribution with parameter q_k.*

In the framework we adopt, the only step we are concerned with in the UNOS allocation policy is the regional allocation step. Our model maximizes the national benefit by redesigning the regions without interfering with any UNOS policy regarding how patients should be prioritized. Hence, once a regional design has been selected, the probability that a patient gets a harvested liver is solely driven by the current policies of UNOS. In other words, there is no societal optimization issue regarding who should get the liver or how patients should be ranked, but we are rather concerned with the design of the regions, which constitutes a very important step in the allocation mechanism, and maximizing the resulting outcome by following UNOS's policies for allocation within the regions. The goal of the second stage is to capture the expected benefit of liver transplants under different scenarios, and second-stage decisions regarding the probabilities of patients on the waiting list being offered a liver are actually automatic, following UNOS's liver allocation regulations, once a first-stage decision has been made. This fact, combined with Assumption 4.3, allows the use of closed-form expressions to calculate patient-transplant probabilities and the second-stage effects of regions designs.

For a candidate region $r \in R$, scenario $\xi^k \in \Xi$ and patient $m \in M_k$, let $j(m|r,k)$ denote the rank of patient m on the regional waiting list for region r. Then, by Assumption 4.3, we can calculate the probability that patient m gets the donated liver in scenario ξ^k as follows:

$$Prob\{m|r,k\} = \begin{cases} (1-q_k)^{j(m|r,k)-1} \cdot q_k, & \text{if } O(k) \in r, \text{ and } j(m|r,k) \leq J_k, \\ 0, & \text{otherwise.} \end{cases} \tag{4.2}$$

Hence, with probability $(1-q_k)^{J_k}$, the liver available under scenario $\xi^k \in \Xi$ will not be used at the regional phase.

Let $w(\xi^k) \in \mathbb{R}^{|M_k|}$ denote the probabilities of the liver being transplanted to the patients on the waiting list under scenario ξ^k. The matching probabilities of organs and patients depend on the regional configuration. Recall that $T(\xi^k)$ gives the matching probabilities for the candidate regions in the first-stage problem. Combining this information with a first-stage solution x, which represents a chosen regional configuration, provides matching probabilities for scenario ξ^k. So, we have

$$w_m(\xi^k) = [T(\xi^k)x]_m = Prob\{\text{patient } m \text{ gets the liver under configuration } x\}.$$

For a given first-stage solution x and a scenario ξ^k, assume $O(k)$ is in region $\hat{r}$ and $x_{\hat{r}} = 1$. Note that although many different regions can contain $O(k)$, only one region variable can be set to 1, due to set-partitioning constraint (4.1b). Then, we can write

$$\begin{aligned} w_m(\xi^k) = [T(\xi^k)x]_m &= Prob\{\text{patient } m \text{ gets the liver under configuration } x\} \\ &= Prob\{m|\hat{r}, k\}. \end{aligned}$$

Hence, the second-stage objective value of regional configuration x under scenario ξ^k can be expressed as follows:

$$Q(x, \xi^k) = c(\xi^k)^T w(\xi^k) = c(\xi^k)^T T(\xi^k)x. \tag{4.3}$$

Let $T_r(\xi^k)$ denote the column of $T(\xi^k)$ for region $r \in R$. Then, following equation (4.3), which provides a closed-form expression for second-stage objective values in terms of x, we can calculate the objective function coefficient of a region r as follows:

$$d_r = \sum_{k=1}^{K} p^k c(\xi^k)^T T_r(\xi^k). \tag{4.4}$$

Hence, we have proven the following proposition:

Proposition 4.1. *The deterministic equivalent program (4.1) is equivalent to*

$$\max \quad \sum_{r \in R} d_r x_r \tag{4.5a}$$

$$\text{subject to} \sum_{r \in R} a_{ir} x_r = 1, \qquad \forall i \in I, \tag{4.5b}$$

$$x_r \in \{0, 1\}, \qquad \forall r \in R. \tag{4.5c}$$

4.2.3 Branch-and-Price Framework

The set of all potential regions, R, is exponential in size and impractical to enumerate explicitly. As we mentioned in Chapter 3, the number of contiguous regions with no more than 8 OPOs is around 3.44×10^5, whereas there are 1.27×10^6 contiguous regions with up to 9 OPOs [96]. For this reason, we adopt a branch-and-price approach in which we utilize a *restricted* set of regions, denoted by R', and generate favorable regions as needed using column generation, which continually updates R'. Note that $R' \subseteq R$ and that all of the definitions above are valid on the R' domain.

In this case, the *Restricted Master Problem (RMP)* becomes:

$$\max \quad \sum_{r \in R'} d_r x_r \tag{4.6a}$$

$$\text{subject to} \sum_{r \in R'} a_{ir} x_r = 1, \qquad \forall i \in I, \tag{4.6b}$$

$$x_r \in \{0, 1\}, \qquad \forall r \in R'. \tag{4.6c}$$

Although the RMP depends on the restricted set of regions R', we drop R' from the notation for ease of exposition.

Pricing Subproblem

Let π_i denote the dual variable corresponding to the set-partitioning constraint for OPO i (4.6b) in the LP relaxation of (4.6). Then, the reduced cost of a region $\hat{r}$ can be calculated as follows:

$$\bar{c}_{\hat{r}} = d_r - \sum_{i \in I} \pi_i a_{i\hat{r}} = \left(\sum_{k=1}^{K} p^k c(\xi^k)^T T_{\hat{r}}(\xi^k) \right) - \sum_{i \in I} \pi_i a_{i\hat{r}}. \tag{4.7a}$$

Thus, we have to find $T_{\hat{r}}(\xi^k)$ for "non-existing" regions, i.e. regions that haven't been generated previously. Let $c_m(\xi^k)$ and $T_{m\hat{r}}(\xi^k)$ denote the m^{th} entries of vectors $c(\xi^k)$ and $T_{\hat{r}}(\xi^k)$. Then, we have

$$\bar{c}_{\hat{r}} = \sum_{k=1}^{K} p^k \left(\sum_{m \in M_k} c_m(\xi^k)\ T_{m\hat{r}}(\xi^k) \right) - \sum_{i \in I} \pi_i a_{i\hat{r}}, \tag{4.8a}$$

$$= \sum_{k=1}^{K} \sum_{m \in M_k} \left(p^k c_m(\xi^k)\ Prob\{m|\hat{r}, k\} \right) - \sum_{i \in I} \pi_i a_{i\hat{r}}. \tag{4.8b}$$

Finding $Prob\{m|\hat{r},k\}$ for a "non-existing" region for all $\xi^k \in \Xi$ and $m \in M_k$ is nontrivial. We introduce a pricing problem that does this by using binary variables for selecting OPOs into a region and ensuring that the ranking of patients conforms to the liver allocation policies in the newly designed region. The objective is to find a region design with the maximum favorable reduced cost that can improve the objective function value of the RMP.

Let y_i be a binary variable such that $y_i = 1$ if OPO $i \in I$ is chosen for the new region, and $y_i = 0$ otherwise, and let $z_{mjk} = 1$ if patient $m \in M_k$ is in j^{th} place in the new region under scenario $\xi^k \in \Xi$, and $z_{mjk} = 0$ otherwise, for $j = 0, \ldots, \min(m, J_k)$. Recall that J_k denotes the number of ranks that matter due to offer limitations related to the CIT of the organ for scenario ξ^k, for $k = 1, \ldots, K$. Hence, for scenario ξ^k, all patients who are not ranked in the first J_k spots are assigned to the 0^{th} place, and consequently, have a zero probability of getting the organ. Note that 0^{th} place is not ahead of the 1^{st} place but shows that a patient is not ranked within the first J_k to be offered the available organ.

For every scenario ξ^k, a patient $m \in M_k$ could be ranked in one of the spots 1 through J_k, if (a) her OPO is chosen for the new region, i.e., $y_{O(m,k)} = 1$, and (b) the donor OPO for scenario ξ^k is in the new region, i.e. $y_{O(k)} = 1$. Thus, we have

$$\sum_{j=1}^{\min(m,J_k)} z_{mjk} \leq y_{O(m,k)}, \qquad k = 1, \ldots, K, \forall m \in M_k, \tag{4.9a}$$

$$\sum_{j=1}^{\min(m,J_k)} z_{mjk} \leq y_{O(k)}, \qquad k = 1, \ldots, K, \forall m \in M_k. \tag{4.9b}$$

Moreover, each of the spots 1 through J_k, for scenario ξ^k, can only be used if and only if the donor OPO for that particular scenario is a part of the new region. Hence,

$$\sum_{m=j}^{|M_k|} z_{mjk} = y_{O(k)}, \qquad k = 1, \ldots, K, j = 1, \ldots, J_k. \tag{4.10}$$

Since we only consider organ-patient matching at the regional phase, for the benefits of a scenario to be realized, the liver available under that scenario has to be in the selected region, and we can only evaluate the benefits of transplant for patients in that particular region.

In addition to these requirements, under scenario ξ^k, a patient $m \in M_k$ has to be either ranked in one of the spots 1 through J_k, or be left unranked. Thus, we have

$$\sum_{j=0}^{\min(m,J_k)} z_{mjk} = 1, \qquad k = 1, \ldots, K, \forall m \in M_k. \tag{4.11}$$

In order to make the constraint set of the pricing problem complete, we need one more set of constraints that force the rankings to conform to UNOS's policies. Recall that during the regional matching phase of the three-tier allocation scheme, the harvested liver is offered to patients listed in the region that contains the donor OPO in decreasing order of sickness. In our modeling framework, the value of the binary vector y determines the set of OPOs that go into the new region. Let $b^k_{mi} = 1$ if $O(m,k) = i$ (i.e., patient m is listed in OPO i under scenario k), and $b^k_{mi} = 0$ otherwise. Also, let $n^k_{mi} = \sum_{l \in M_k : l \leq m} b^k_{li}$, so that n^k_{mi} shows the number of patients listed in OPO i who are ranked above or at the same spot as patient m (including patient m herself) on the *national waiting list*, i.e., across the whole country, under scenario ξ^k. Thus, $\sum_{i \in I} n^k_{mi} y_i$ is the number of people at least as sick as patient m under scenario ξ^k that are listed in the new region, which is equal to the *regional rank* of patient m under scenario ξ^k. The following set of constraints relies on this observation, and together with constraints (4.9a), (4.9b), (4.10) and (4.11), enforces the regional ranks to be assigned in decreasing order of illness severity within the region:

$$\sum_{j=1}^{\min(m,J_k)} j \cdot z_{mjk} + m \cdot z_{m0k} \geq \sum_{i \in I} n^k_{mi} \cdot y_i, \qquad k = 1, \ldots, K, \forall m \in M_k. \tag{4.12}$$

The system of inequalities described by (4.9)-(4.12) forms the constraint set of our pricing problem. A very important observation is that for a fixed binary vector y, this system is separable for each scenario $k = 1, \ldots, K$. Table 4.1 shows the structure of the constraint matrix formed by the system (4.9)-(4.12). Each row in the table corresponds to the coefficient matrix for an individual scenario and the thick dots represent variable groups that have nonzero coefficients for the constraint sets of that particular scenario. From Table 4.1, it is easy to see that once the y variables are fixed, we are left with K independent subsystems, each of which corresponds to an individual scenario in Ξ.

Table 4.1: Constraint structure of the system of inequalities represented by (4.9)-(4.12).

Scenario	Variables							
	y	z_{mj1}	z_{m01}	z_{mj2}	z_{m02}	$\cdots$	z_{mjK}	z_{m0K}
1	•	•						
	•	•						
	•	•						
	•	•	•					
	•	•	•					
2	•			•				
	•			•				
	•			•				
	•			•	•			
	•			•	•			
$\vdots$						$\ddots$		
K	•						•	
	•						•	
	•						•	
	•						•	•
	•						•	•

For a fixed $y \in \mathbb{B}^{|I|}$ and scenario ξ^k, let $P(y,k) = \{z|(4.13a) - (4.13g)\}$, where

$$\sum_{j=1}^{\min(m,J_k)} z_{mj} \leq y_{O(m,k)}, \qquad \forall m \in M_k, \tag{4.13a}$$

$$\sum_{j=1}^{\min(m,J_k)} z_{mj} \leq y_{O(k)}, \qquad \forall m \in M_k, \tag{4.13b}$$

$$\sum_{m=j}^{|M_k|} z_{mj} = y_{O(k)}, \qquad j = 1, \ldots, J_k, \tag{4.13c}$$

$$\sum_{j=0}^{\min(m,J_k)} z_{mj} = 1, \qquad \forall m \in M_k, \tag{4.13d}$$

$$\sum_{j=1}^{\min(m,J_k)} j \cdot z_{mj} + m \cdot z_{m0} \geq \sum_{i \in I} n_{mi}^k \cdot y_i, \qquad \forall m \in M_k, \tag{4.13e}$$

$$z_{m0} \in \{0,1\}, \qquad \forall m \in M_k, \tag{4.13f}$$

$$0 \leq z_{mj} \leq 1, \qquad \forall m \in M_k, j = 1, \ldots, J_k. \tag{4.13g}$$

Now, we will prove that a solution to the system $P(y,k)$ ranks the patients under scenario ξ^k in decreasing order of sickness in the region defined by y. In order to prove this claim, we need the following set of lemmas.

Lemma 4.1. *For a fixed $y \in \mathbb{B}^{|I|}$ and scenario $\xi^k \in \Xi$,*

(a) if $y_{O(k)} = 0$, then $z_{mj} = 0$ and $z_{m0} = 1, \forall m \in M_k,\ j = 1, \ldots, \min(m, J_k)$,

(b) if $y_{O(k)} = 1$, then for a patient $m \in M_k$ such that $y_{O(m,k)} = 0$, $z_{m0} = 1$ and $z_{mj} = 0$, for $j = 1, \ldots, \min(m, J_k)$.

Proof. If $y_{O(k)} = 0$, then by (4.13b) and (4.13g), $z_{mj} = 0, \forall m \in M_k,\ j = 1, \ldots, \min(m, J_k)$. Hence, (4.13d) forces $z_{m0} = 1, \forall m \in M_k$, which implies that none of the patients in set M_k will be offered the liver harvested under scenario ξ^k at the regional level.

If $y_{O(k)} = 1$, for a patient $m \in M_k$ such that $y_{O(m,k)} = 0$, by (4.13a) and (4.13g), $z_{mj} = 0, j = 1, \ldots, \min(m, J_k)$. So, by constraint (4.13d), $z_{m0} = 1$ is forced. Hence, patients that are not listed in the region defined by y cannot be offered the liver harvested under scenario ξ^k at the regional level. □

Lemma 4.2. *For a scenario $\xi^k \in \Xi$, patient $m \in M_k$ and an integer $1 \leq \ell \leq \min(m, J_k)$,*

(a) $j' = \left\{\max \sum_{j=1}^{\ell} j \cdot z_{mj} | \sum_{j=0}^{\ell} z_{mj} = 1, z_{mj} \geq 0, j = 0, \ldots, \ell\right\} = \ell$, *and*

(b) $j'' = \left\{\max \sum_{j=1}^{\ell} j \cdot z_{mj} + m \cdot z_{m0} | \sum_{j=0}^{\ell} z_{mj} = 1, z_{mj} \geq 0, j = 0, \ldots, \ell\right\} = m$.

Proof. The proof is straightforward. Here we just illustrate it for part *(a)*. Note that the variable with the biggest objective function coefficient is $z_{m\ell}$, and the upper bound of $z_{m\ell}$ is 1. Setting $z_{m\ell} = 1$ gives the maximum objective value, i.e., $j' = \ell$. □

Lemma 4.3. *For an OPO vector $y \in \mathbb{B}^{|I|}$, scenario $\xi^k \in \Xi$ and patient $m \in M_k$, $m \geq \sum_{i \in I} n^k_{mi} y_i$.*

Proof. Since $y_i \leq 1, \forall i \in I$, we have $\sum_{i \in I} n^k_{mi} y_i \leq \sum_{i \in I} n^k_{mi}$. By using the definition of n^k_{mi}, $\sum_{i \in I} n^k_{mi} = \sum_{i \in I} \sum_{l \in M_k : l \leq m} b^k_{li}$. Since a patient can only be listed in one OPO by Assumption 4.2, we know that $\sum_{i \in I} b^k_{li} = 1$. Thus, by rearranging terms, we have $\sum_{i \in I} \sum_{l \in M_k : l \leq m} b^k_{li} = \sum_{l \in M_k : l \leq m} \sum_{i \in I} b^k_{li} = \sum_{l \in M_k : l \leq m} 1 = m$. □

For $y \in \mathbb{B}^{|I|}$ and $k = 1, \ldots, K$, let $m(y, \ell)$ denote the index $m' \in M_k$ such that $\sum_{i \in I} n^k_{m'i} y_i = \ell$ where $M(y, k) = \{m \in M_k | y_{O(m,k)} = 1\}$, i.e., the set of patients in the region defined by y. Note that $m(y, \ell)$ shows the index of the patient whose regional UNOS rank is ℓ under scenario ξ^k. Although $m(y, \ell)$ also depends on scenario ξ^k, we drop k from the notation for ease of exposition.

Lemma 4.4. *For $J_k < \ell \leq |M(y, k)|$, $z_{m(y,\ell)0} = 1$ and $z_{m(y,\ell)j} = 0$, for $j = 1, \ldots, \min(m(y, \ell), J_k)$ under scenario $\xi^k \in \Xi$.*

Proof. By Lemma 4.2, $\max \sum_{j=1}^{J_k} j \cdot z_{m(y,\ell)j} = J_k$. Since $\ell > J_k$, (4.13e) cannot be satisfied by setting $\sum_{j=0}^{J_k} z_{m(y,\ell)j} = 1$. However, since $m(y, \ell) \geq \ell$ by Lemma 4.3, setting $z_{m(y,\ell)0} = 1$ will make sure that both (4.13e) and (4.13d) are satisfied. □

Now, consider the case where $\ell \leq J_k$.

Lemma 4.5. *Suppose $y_{O(k)} = 1$ for scenario $\xi^k \in \Xi$. Then, for $\ell \leq J_k$, (a) $z_{m(y,\ell)0} = 0$, and (b) $z_{m(y,\ell)\ell} = 1$, and $z_{m(y,\ell)j} = 0$ for $j = 1, \ldots, \min(m(y, \ell), J_k), j \neq \ell$.*

Proof. For the sake of simplicity in notation, assume $m(y,\ell) \geq J_k$. Suppose for some $m(y,\ell') = m' \in M(y,k)$ where $\ell' \leq J_k$, $z_{m'0} = 1$. Then, by (4.13d), $\sum_{j=1}^{J_k} z_{m'j} = 0$. Summing up constraint set (4.13c) over the spots that have to be filled, i.e., $j = 1,\ldots,J_k$, yields

$$\sum_{j=1}^{J_k} \sum_{m=j}^{|M_k|} z_{mj} = J_k. \tag{4.14}$$

By Lemma 4.1, $z_{mj} = 0, \forall m \notin M(y,k)$, and by Lemma 4.4 and (4.13d), $z_{m(y,\ell)j} = 0$ for $\ell = J_k + 1,\ldots,|M(y,k)|$. Thus, (4.14) becomes

$$\sum_{j=1}^{J_k} \sum_{\ell=1}^{J_k} z_{m(y,\ell)j} = J_k. \tag{4.15}$$

Changing the order of the summation terms in (4.15) gives

$$\sum_{\ell=1}^{J_k} \sum_{j=1}^{J_k} z_{m(y,\ell)j} = J_k. \tag{4.16}$$

Note that (4.16) can only hold if $\sum_{j=1}^{J_k} z_{m(y,\ell)j} = 1$, for $\ell = 1,\ldots,J_k$. However, we have already shown that for ℓ', $\sum_{j=1}^{J_k} z_{m'j} = 0$. Thus, by contradiction, $z_{m(y,\ell)0} = 0, \forall m(y,\ell) \in M(y,k)$ where $\ell \leq J_k$. This proves part *(a)*.

By part *(a)*, for $\ell \leq J_k$, $z_{m(y,\ell)0} = 0$. Thus, for $\ell = 1,\ldots,J_k$, constraint (4.13e) becomes

$$\sum_{j=1}^{J_k} j \cdot z_{m(y,\ell)j} \geq \ell. \tag{4.17}$$

For $\ell = J_k$, by (4.17) and Lemma 4.2, $z_{m(y,J_k)J_k} = 1$. From (4.13d) and (4.13c), it follows that $z_{m(y,J_k)j} = 0$ for $j = 1,\ldots,J_k - 1$, and $z_{m(y,\ell)J_k} = 0$ for $\ell = 1,\ldots,J_k - 1$. Assume the claim holds for $\ell = l,\ldots,J_k$. Then, for $\ell = l-1$, constraint (4.13e) reduces to

$$\sum_{j=1}^{l-1} j \cdot z_{m(y,l-1)j} \geq l - 1. \tag{4.18}$$

Constraint (4.18) is only satisfied if $z_{m(y,l-1)l-1} = 1$. From (4.13d) and (4.13c), it follows that $z_{m(y,l-1)j} = 0$ for $j = 1,\ldots,J_k, j \neq l-1$, and $z_{m(y,\ell)l-1} = 0$ for $\ell = 1,\ldots,J_k, \ell \neq l-1$. Hence, by induction, part *(b)* is proven. The case where $m(y,\ell) < J_k$ can be shown in a similar fashion. □

We are now ready to state Theorem 4.1. The proof follows from Lemmas 4.1 through 4.5.

Theorem 4.1. *For a fixed OPO vector $y \in \mathbb{B}^{|I|}$ and scenario $\xi^k \in \Xi$, a solution to the system $P(y,k)$ ranks the patients under scenario ξ^k in decreasing order of sickness in the region defined by y.*

We have already discussed that the scenario constraint matrices are separable once a fixed y is given (see Table 4.1). Hence, the following corollary follows:

Corollary 4.1. *For a fixed OPO vector $y \in \mathbb{B}^{|I|}$, a solution to the system*

$$P(y) = \{z_{m0k} \in \{0,1\}, 0 \leq z_{mjk} \leq 1, \text{ for } k = 1, \ldots, K, m \in M_k,$$

$$j = 1, \ldots, \min(m, J_k) | (4.9) - (4.12)\},$$

ranks the patients under all scenarios in Ξ in decreasing order of sickness in the region defined by y.

Now, let us focus on the objective function of the pricing problem. For $k = 1, \ldots, K$ and $m \in M_k$, let

$$\tilde{c}_{mjk} = \begin{cases} (1-q_k)^{j-1} \cdot q_k \cdot c_m(\xi^k), & \text{if } 1 \leq j \leq J_k, \\ 0, & \text{otherwise.} \end{cases} \tag{4.19}$$

The parameter $\tilde{c}_{mjk}$ can be interpreted as the contribution of patient m to the expected benefit of liver transplantation for scenario ξ^k if the patient is ranked j^{th}. For every scenario $\xi^k \in \Xi$, by Assumption 4.3, the rank of the patient that will receive the transplantation has a geometric distribution with parameter q_k. So, if patient m is ranked in the j^{th} spot then by this assumption, the probability that patient m receives the liver will be $(1-q_k)^{j-1} \cdot q_k$, and the benefit that this patient gets will be $c_m(\xi^k)$.

By using $\tilde{c}_{mjk}$, (4.8b) becomes

$$\sum_{k=1}^{K} \sum_{m \in M_k} \sum_{j=1}^{\min(m, J_k)} p^k \cdot \tilde{c}_{mjk} \cdot z_{mjk} - \sum_{i \in I} \pi_i \cdot y_i. \tag{4.20}$$

The pricing problem needs to be able to handle the ranking of patients for different regions for every scenario, and needs to incorporate the patient-acceptance probabilities that are dependent on these rankings. The following formulation manages this:

$$SRPP(\pi, I, K) = \max \sum_{k=1}^{K} \sum_{m \in M_k} \sum_{j=1}^{\min(m,J_k)} p^k \cdot \tilde{c}_{mjk} \cdot z_{mjk} - \sum_{i \in I} \pi_i \cdot y_i \tag{4.21a}$$

$$\text{subject to} \sum_{j=1}^{\min(m,J_k)} z_{mjk} \leq y_{O(m,k)}, \qquad k = 1, \ldots, K, \forall m \in M_k, \tag{4.21b}$$

$$\sum_{j=1}^{\min(m,J_k)} z_{mjk} \leq y_{O(k)}, \qquad k = 1, \ldots, K, \forall m \in M_k, \tag{4.21c}$$

$$\sum_{m=j}^{|M_k|} z_{mjk} = y_{O(k)}, \qquad k = 1, \ldots, K, j = 1, \ldots, J_k, \tag{4.21d}$$

$$\sum_{j=0}^{\min(m,J_k)} z_{mjk} = 1, \qquad k = 1, \ldots, K, \forall m \in M_k, \tag{4.21e}$$

$$\sum_{j=1}^{\min(m,J_k)} j \cdot z_{mjk} + m \cdot z_{m0k} \geq \sum_{i \in I} n^k_{mi} \cdot y_i, \qquad k = 1, \ldots, K, \forall m \in M_k, \tag{4.21f}$$

$$y_i \in \{0, 1\}, \qquad \forall i \in I, \tag{4.21g}$$

$$z_{m0k} \in \{0, 1\}, \qquad k = 1, \ldots, K, \forall m \in M_k, \tag{4.21h}$$

$$0 \leq z_{mjk} \leq 1, \qquad k = 1, \ldots, K, j = 1, \ldots, J_k, \forall m \in M_k. \tag{4.21i}$$

The objective function (4.21a) maximizes the reduced cost of a region design. Constraint set (4.21b) ensures that patients can only be ranked if they are listed in an OPO that is chosen for new region design. Constraint (4.21c) makes sure that patients are ranked only for scenarios that apply to the new region, i.e., scenarios under which a liver is harvested within the region. Constraint set (4.21d) forces every spot from 1 to J_k to be assigned a patient only if scenario ξ^k has a harvested organ within the new region, and no ranks to be used otherwise. Constraint (4.21e) makes sure that each patient is either assigned to one of the spots 1 through J_k or left unranked. Constraint set (4.21f) enforces the patients to be ranked in descending order of the severity of illness. And finally, constraints (4.21g) and (4.21h) force variables y_i and z_{m0k} to take on binary values, respectively, while constraint (4.21i) determines the range for variables z_{mjk}.

Note that the pricing problem (4.21) is a two-stage stochastic mixed-mixed integer program. The deterministic equivalent of (4.21) can be formulated as follows:

$$SRPP(\pi, I, K) = \max \mathbb{E}_{\xi} \tilde{Q}(y, \xi^k) - \sum_{i \in I} \pi_i \cdot y_i \tag{4.22a}$$

$$\text{subject to} \quad y_i \in \{0, 1\}, \qquad \forall i \in I, \tag{4.22b}$$

where

$$\tilde{Q}(y, \xi^k) = \left\{ \max \sum_{m \in M_k} \sum_{j=1}^{\min(m, J_k)} \tilde{c}_{mj} z_{mj} | (4.13a) - (4.13g) \right\}, \quad \text{for } k = 1, \ldots, K. \tag{4.23}$$

Suppose we solve the pricing problem (4.21) to optimality, and obtain the optimal solution vector (y^*, z^*). Then, y^* gives the region design with maximum reduced cost. Let us denote this region by $\hat{r}$. If the objective value (4.21a) is strictly positive, we add $\hat{r}$ to the restricted set of regions, R', and the column that corresponds to region $\hat{r}$ in formulation (4.6) can be constructed as follows:

- Objective function (4.6a) coefficient for region $\hat{r}$: $d_{\hat{r}} = \sum_{k=1}^{K} p^k \left(\sum_{m \in M_k} \tilde{c}_{mjk} \cdot z^*_{mjk} \right)$,
- For constraint set (4.6b), $a_{i\hat{r}} = y^*_i, \quad \forall i \in I$.

Remark 4.1. *Perhaps the most important effect of Theorem 4.1 and Corollary 4.1 is that they show that variables z_{mjk} do not have to be binary, but can simply be continuous for $k = 1, \ldots, K, m \in M_k, j = 1, \ldots, \min(m, J_k)$. For instance, suppose $J_k = J$ for $k = 1, \ldots, K$. Then, the potential number of these variables is $O(K \cdot J^2 \cdot |I|)$, and being able to prove that this many variables are actually **implied binaries** that do not have to branched on has an immense effect on the computational tractability of the problem.*

4.3 COMPUTATIONAL APPROACHES

In Section 4.2.3, we described the details for an *ideal* solution procedure, with a fixed number of scenarios that capture the underlying uncertainty in the national waiting list and a pricing problem solved for the whole nation, explicitly considering every scenario, that generates promising regions throughout the branch-and-bound tree. However, the resulting formulation poses significant computational challenges and is very hard to solve to optimality.

First, recall that the scenarios include information on the organs donated, donor characteristics and OPOs, patient populations with respect to different geographic locations, and patient characteristics. Hence, these sources of uncertainty in the system would lead to potentially infinitely many scenarios to fully capture *every possible state* of the national liver allocation system.

Secondly, the pricing problem formulation (4.21) that was discussed in the previous section constitutes a very large-scale mixed-integer program that is hard to solve. In addition to this, the pricing problem has to be solved over and over again throughout the solution process. Further simplifications are needed to make this practical.

In this section we discuss our computational approaches on how to solve the large-scale integer program that we introduced. We group our discussion under three main titles: scenario generation, column generation, and the pricing problem.

4.3.1 Computational Approaches for Scenario Generation

As there are too many scenarios in the real-world system to consider explicitly from a modeling and solution perspective, in this chapter, we explore methods that involve *Monte Carlo sampling* procedures.

Each function value in a stochastic program can involve a multidimensional integral with extremely high dimensions. Because Monte Carlo simulation appears to offer the best possibilities for higher dimensions [48], it seems to be the natural choice for use in stochastic programs [23]. Monte Carlo approaches use samples from the underlying scenario distribution to approximate the stochastic behavior of the system. An important feature is using statistical estimates to obtain confidence intervals of results.

A popular technique involving Monte Carlo sampling is *External Sampling*, which is also known as the *Sample Average Approximation* (SAA), or *Sample-Path Optimization* method. The basic idea in external sampling is to take a sample of K scenarios and solve the resulting stochastic program that is based on this fixed sample to optimality. After repeating the procedure a number of times using independent samples until a stopping criterion is met, it is possible to derive statistical properties and confidence intervals around the quality of the solution.

Using external sampling to construct and solve the sample average approximation problem for stochastic programs has been a widely used idea in the area of stochastic programming. Many authors have explored statistical behavior of solutions for this method and used the framework in a wide variety of applications [23, 95, 122, 154].

We now describe how the *Sample Average Approximation* (SAA) problem is formulated for our stochastic region design problem. Let d_r^K denote the objective function coefficient of a region r for a sample of K scenarios. Then,

$$d_r^K = \frac{1}{K} \sum_{k=1}^{K} c(\xi^k)^T T_r(\xi^k). \tag{4.24}$$

Hence, under the SAA, the RMP (4.6) is replaced with the following (RMP$_{SAA}$):

$$\max \hat{g}_K(x) = \sum_{r \in R'} d_r^K x_r \tag{4.25a}$$

$$\text{subject to} \quad (4.6b) - (4.6c), \tag{4.25b}$$

and (4.25) is solved B times using B independent samples of K scenarios.

4.3.2 SPRINT Approach for Column Generation

As we discussed before, the sheer size of the pricing problem itself poses difficulties in the solution procedure. We borrow the notion of SPRINT methods from airline crew scheduling literature [12], which involve extensive pricing *only* in the root node of a branch-and-bound tree to aid us in the solution process. In a SPRINT procedure, the problem is loaded with a set of initial columns (potentially thousands), and the LP is optimized over those columns. Then, a pricing process is carried out where thousands of more columns are added, and most of the nonbasic columns are discarded. The process is repeated until all columns for the LP have been considered. Then, the IP solution procedure starts with the current set of columns in the formulation without any further pricing throughout the solution process. SPRINT approaches have been very successful in solving large-scale airline crew-pairing problems [12, 14], which are also set-partitioning based formulations, like our integer programming formulation (4.5). These approaches have been widely studied in the literature [9, 24, 82].

Once again, we make use of the *Geographic Decomposition* approach of Kong [96] and Kong et al. [97], as we did in Chapter 3. Thus, instead of considering a very large pricing problem for the whole nation, we create a number of smaller pricing problems, each covering different, but overlapping, geographic areas of the country, called *region covers*. Suppose we have a set of region covers denoted by $\mathcal{C}$. Let I_j denote the set of OPOs in region cover $j \in \mathcal{C}$, and let K_j denote the set of scenarios in which the donor OPO is within region cover $j \in \mathcal{C}$. Thus, $K_j = \{k = 1, \ldots, K | O(k) \in I_j\}$. Let us denote the associated pricing problem for region cover $j \in \mathcal{C}$ by $SRPP(\pi, I_j, K_j)$. Replacing the terms I, K and M_k with I_j, K_j and $M_{k,j}$ in (4.21), where $M_{k,j} = \{m \in M_k | O(m, k) \in I_j\}$, gives the description of the pricing problem for region cover $j \in \mathcal{C}$, i.e., $SRPP(\pi, I_j, K_j)$. Hence, using geographic decomposition not only affects the OPO set in a pricing problem, but also scales down the scenario and patient sets associated with the pricing problem, which makes the problem computationally more tractable.

We also employ the branching strategy of Ryan and Foster [148] which has proven to be effective on set-partitioning type problems. See Sections 3.3.1.1 and 3.3.1.2 for further details on these techniques.

4.3.3 Solution Methods for the Pricing Problem

In order to be able to solve the pricing problem (4.21) effectively, we exploit the special structure of the problem, and employ various branching, heuristic solution generation, and pricing methods.

4.3.3.1 Branching Routines A very important observation about the pricing problem structure is that once y is fixed, the ranking of patients under different scenarios is the only feasible solution. The composition of the new region defined by the binary vector y is the only factor that drives how patients are ranked, due to the fact that we strictly model UNOS's ranking policies for the regional phase. Of course, variables y_i are not the only variables in the pricing problem. In fact, they only constitute a very small number, i.e., $|I|$, when compared to the total number of variables. Still, during the branch-and-bound process, how the values and bounds of y change throughout the solution procedure gives us valuable information to be used in generating effective bounds for the binary variables z_{m0k} and fixing them, which enables us to construct effective branching patterns.

Before laying out the details of our branching routine for y variables, we present the following observations:

- *Down branch on variable y_i*

  In the child node created by a down branch on some OPO variable y_i none of the patients listed in OPO i can be ranked, due to constraint (4.9a).

- *Up branch on variable y_i*

  Setting $y_i = 1$ in the child-node resulting from an up branch will ensure that none of the patients who are below the J_k^{th} patient listed in OPO i can be ranked. Since OPO i is being forced to be in the new region, the top-most J_k patients listed in OPO i will be candidates for spots 1 through J_k to get an organ offer. Since we limit the number of regional offers to J_k, no patient below the J_k^{th} patient listed in OPO i can be offered the organ. Hence, the corresponding z_{m0k} variables for these patients can be set to 1 for this up branch. In fact, a generalization of this idea is possible where we look at *all* OPOs that are being forced into the region at an up branch and force a number patients to

be left unranked by counting the patient population in these enforced OPOs until the counter hits J_k. While traversing the patient population in decreasing order of sickness, none of the remaining patients, regardless of where they are listed, can be ranked once the counter hits J_k, because this means that J_k transplant candidates have already been identified in the region.

In this section, we give a branching routine that will be used every time an OPO variable is selected to be branched on, during branch and bound. This routine kicks in after an OPO variable has been identified and before creating new nodes as a result of branching. The decision regarding which variable to branch on depends on variable selection strategy utilized during the branch-and-bound process, i.e., pseudo-costs, strong branching, etc. Let $y_{i'}$ denote the variable chosen to be branched on.

Branching on an OPO variable $(y_{i'})$

1. *Down branch.* For the newly created child node,
   - Set $y_{i'} = 0$.
   - Set $z_{m0k} = 1$, for $k = 1, \ldots, K, m \in M_k$ such that $O(m, k) = i'$.
2. *Up branch.* Let $LB(y_i)$ denote the lower bound of variable y_i in the current node.
   - Define $C = \{i'\} \cup \{i \in I | LB(y_i) = 1\}$.
   - Define $M_k(C, l) = \{m \in M_k | O(m, k) \in C, m \leq l\}$.
   - Find m' such that $|M_k(C, m')| = J_k$
   - For the newly created child node,
     - Set $y_{i'} = 1$.
     - Set $z_{m0k} = 1$, for $k = 1, \ldots, K, m \in M_k$ such that $m > m'$.

Another important observation about the pricing problem structure is that once $z_{m'0k}$ is fixed for some $k, m' \in M_k$, this implies that a number of patient-unranked variables can also be fixed. For instance, if $z_{m'0k} = 0$, this implies $z_{m0k} = 0$ for $m < m'$ such that $O(m, k) = O(m', k)$. Similarly, setting $z_{m'0k} = 1$ implies $z_{m0k} = 1$ for $m > m'$ such that $O(m, k) = O(m', k)$. A generalization similar to the one we utilized for y variables is also possible.

We also give a branching routine that will be used every time a patient-unranked variable is selected to be branched on, throughout the branch-and-bound process. Let $z_{m'0k}$ denote the variable chosen to be branched on.

Branching on a patient-unranked variable **($z_{m'0k}$)**

1. *Down branch.* Let $LB(y_i)$ denote the lower bound of variable y_i in the current node.
   - Find $C = \{i \in I | LB(y_i) = 1\}$.
   - If $O(m', k) \notin C$, define $C' = C \cup O(m', k)$, otherwise set $C' = C$.
   - For the newly created child node,
     - Set $z_{m'0k} = 0$.
     - If $O(m', k) \notin C$, set $y_{O(m',k)} = 1$.
     - Set $z_{m0k} = 0$, for $m \in M_k$ such that $O(m, k) \in C', m < m'$.
2. *Up branch.* For the newly created child node,
   - Set $z_{m'0k} = 1$.
   - If $O(m', k) \in C$, set $z_{m0k} = 1$, for $m \in M_k$ such that $O(m, k) \in C, m > m'$.
     Otherwise, $z_{m0k} = 1$, for $m \in M_k$ such that $O(m, k) = O(m', k), m > m'$.

In addition to our branching routines on variables, we used branching priorities and a variable selection technique known as strong branching. We prioritized y variables over the variables z_{m0k}, so that whenever a fractional solution is found in a node of the branch-and-bound tree, if at least one y variable is fractional, no z_{m0k} can be branched on. In other words, a fractional z_{m0k} variable can be chosen for branching only if all y variables are integers in the current node solution.

Strong branching is a variable selection technique that became available in CPLEX 7.5 [84]. The idea behind strong branching is to solve a number of subproblems partially with tentative branches to see which branch is the most promising before actually branching on a candidate variable [83]. Thus, temporary child nodes are created for each candidate variable, and a number of simplex iterations are performed to evaluate the progress. Although strong branching significantly reduces the number of nodes explored in a branch-and-bound tree, it increases the time spent on each node when compared to other branching techniques. Hence,

its effect on the total solution time is problem dependent, and in some large-scale MIPs it has proven to be very effective.

In addition to these, we employed a multiple-pricing approach and added all favorable regions, instead of just the optimal solution, generated during the solution of a pricing problem to the RMP.

4.3.3.2 Integer Solution Heuristic for Branch-and-Bound

As mentioned previously, once an integral y is obtained, the rankings of patients can be calculated automatically. In this section, we introduce the steps to generate integer solutions from integral y values or by rounding y variables.

Integer Solution Heuristic

1. Define $I(y) = \{i \in I | y_i > \varepsilon_y\}$, where ε_y denotes the threshold parameter for rounding.
2. Set $y_i = 1$ for $i \in I(y)$, and $y_i = 0$ for $i \notin I(y)$.
3. For $k = 1, \ldots, K$,
   - Find m' such that $|M_k(I(y), m')| = J_k$.
   - Let $m_{(l)}$ denote the l^{th} sickest patient in set $M_k(I(y), m')$.
   - For $j = 1, \ldots, J_k$,
     - set $z_{m_{(j)}jk} = 1$ and $z_{m_{(j)}0k} = 0$,
     - set $z_{m_{(j)}sk} = 0$ for $s = 1, \ldots, \min(m(j), J_k), s \neq j$.
   - For $m \notin M_k(I(y), m')$, set $z_{m0k} = 1$, and $z_{mjk} = 0$ for $j = 1, \ldots, \min(m, J_k)$.
4. Solution (y, z) is an integer solution for the pricing problem.

In our implementation, we used $\varepsilon_y = 0.5$ and calculated $I(y)$ in every node. We executed steps 2 through 4 only if $I(y)$ constituted a different region then regions found earlier during the branch-and-bound process.

4.4 DATA SOURCES AND PARAMETER ESTIMATION

The waiting list data for scenarios was generated using the discrete-event simulation of End-Stage Liver Disease and organ allocation by Shechter et al. [155]. We modified the simulation model to capture snapshots of the national liver waiting list, and created scenarios using a sampling approach.

We generated scenarios in the following manner. National patient and organ arrival rates in the simulation model, which are actually dependent on the simulation year, were averaged over the period from the beginning of 1999 to the end of 2002. In order to generate B independent and identically distributed batches of K scenarios, we ran the simulation with B different random seeds until it reached steady state, and recorded all relevant information regarding the national waiting list at the point of transition into a steady state. The simulation was preloaded with the steady-state conditions and was replicated K times for one more year using K different random seeds for each batch. After one year into the steady state, a replication was stopped whenever a liver became available at the regional stage, and a scenario was created using the waiting list data at that point. Patient/liver related data, like MELD scores, locations and expected benefit of the available liver for each patient, was extracted from the terminal waiting list. Figure 4.1 depicts the sampling approach we used to create discrete scenarios using the simulation model of Shechter et al. [155].

The number of patients that potentially can be offered the liver at the regional matching phase, i.e., J_k for each scenario, was set to 10 across all scenarios. There are a couple of reasons behind this. First, as mentioned in Section 1, although the medically acceptable CIT for an organ can potentially go up to 18-24 hours [39], most livers have a CIT less than 10 hours [171]. By policy, a transplant team, i.e., a potential organ recipient and the transplant coordinator, surgeon and physician assigned to her, has only one hour to make a decision about an organ offer [170]. Hence, for an average liver, setting $J_k = 10$ could be considered to be sufficient. Furthermore, since the size of the pricing problems in our modeling framework is sensitive to the number of candidates that have to be considered in every OPO, $J_k > 10$ would significantly increase our computational time.

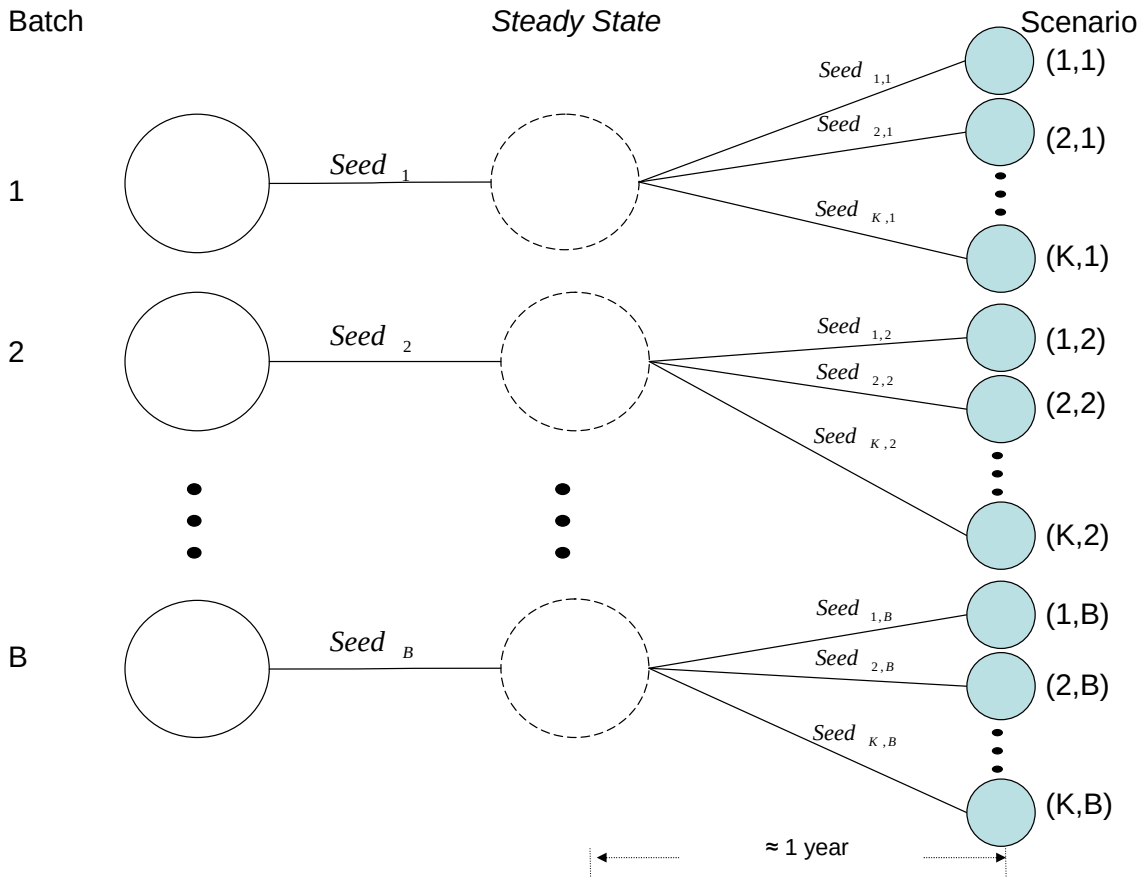


Figure 4.1: Scenario sampling from the End-stage Liver Disease and organ allocation simulation model of Shechter et al. [155].

We also used a constant value for a patient's probability of accepting a liver offer, i.e., q_k, across all scenarios. As mentioned in earlier sections, in the three-tier liver allocation mechanism, the number transplants that occur at the intra-regional level is roughly around 50% of all transplants while national-level transplants account for about 5% [169]. This would be comparable to a case where 10% of the organs that become available to the regional level end up not being used at the regional level. In our modeling framework, an available organ at a certain scenario ξ^k, for $k = 1, \ldots, K$, would not be transplanted at the regional phase if the top J_k candidates in the region reject it. Since we set $J_k = 10$, by Assumption 4.3, the

probability of an organ not being used at the regional level could be calculated as

$$Prob\{\text{Liver is not used at the regional phase under scenario } \xi^k\} = 1 - \sum_{j=1}^{10}(1-q_k)^{j-1}q_k, \tag{4.26}$$

and setting (4.26) to 0.10 yields $q_k = 0.206$, for $k = 1, \ldots, K$.

4.5 COMPUTATIONAL RESULTS

For our computational experiments, we used geographic decomposition scheme 20_10, which was one of the schemes designed and used in Chapter 4. We generated 10 batches, each containing 200 scenarios. We designed two sets of experiments: in one set, we solved these 10 batches using default CPLEX parameters during optimization, and in the other, we used the computational approaches that were introduced in Section 4.3. The initial set of regions in each solution approach was the set of all enumerated regions with no more than 3 OPOs. We used a UNIX machine with AMD Opteron 240 processor and 3.8 GB RAM for our computational tests, and coded the optimization algorithms using C++ and the CPLEX 9.0 Callable Library.

Table 4.2 shows the CPU times for the proposed solution approach versus using CPLEX with default settings. Although our branching and integer solution generation schemes bring computational savings of about 35% on the average when compared to the default CPLEX settings, the average solution time for a batch of 200 scenarios was more than 4 hours. This is mainly due to the size of the pricing problems. Table 4.3 displays the number of rows, columns and nonzeros in every pricing problem for the geographic decomposition scheme 20_10 for the first batch of scenarios. Although the numbers will be different for other samples, since the number of scenarios is equal among all batches, they will still be comparable.

Table 4.2: Results for initial computational experiments using CPLEX: Default setting versus the proposed approach.

Batch	Total CPU time (hour:min:sec)		Savings in CPU time
	Default settings	Proposed approach[a]	
1	5:40:40	4:21:41	23.19%
2	6:44:05	3:51:13	42.78%
3	6:51:08	5:03:49	26.10%
4	6:21:52	3:26:24	45.95%
5	6:08:09	3:24:31	44.45%
6	6:46:50	3:45:43	44.52%
7	6:48:15	4:04:41	40.07%
8	7:06:51	5:23:16	24.26%
9	5:28:13	3:44:34	31.58%
10	6:40:50	4:58:17	25.58%
average	6:27:41	4:12:25	34.85%

[a]Using the branching and rounding routines described in Section 4.3.3 along with CPLEX.

Table 4.3: Size of pricing problems for geographic decomposition scheme 20_10.

Region Cover	Rows	Columns			Nonzeros
		Binary	Continuous	Total	
1	11,855	1,784	19,335	21,119	140,474
2	8,127	1,218	13,235	14,453	95,915
3	14,396	2,198	23,585	25,783	173,175
4	13,444	1,980	21,790	23,770	156,709
5	7,852	1,130	12,630	13,760	90,036
6	15,730	2,392	25,745	28,137	189,423
7	8,481	1,293	13,875	15,168	101,843
8	12,044	1,840	19,730	21,570	145,221
9	18,617	2,840	30,500	33,340	224,446
10	10,334	1,599	16,990	18,589	126,948
11	15,642	2,385	25,620	28,005	188,906
12	14,702	2,242	24,080	26,322	177,343
13	10,224	1,564	16,750	18,314	123,006
14	16,624	2,535	27,230	29,765	199,724
15	12,958	2,003	21,305	23,308	158,570
16	13,860	2,140	22,785	24,925	169,512
17	13,036	1,992	21,360	23,352	158,290
18	20,461	3,120	33,520	36,640	246,768
19	14,979	2,313	24,625	26,938	183,820
20	16,600	2,531	27,190	29,721	199,849
average	13,498	2,055	22,094	24,149	162,499

As can be seen from Table 4.3, the pricing problems for the first sample of 200 scenarios under geographic decomposition scheme 20_10 are large-scale mixed-integer programs with around 13,000 rows and 24,000 variables (out of which 2,000 are binaries). The size of the pricing problems makes the model impractical to test for more scenarios or geographic decomposition schemes with larger region covers.

With a limited number of scenarios, this method may also be sensitive to outlier patients that may exist within the samples. Hence, the optimal solutions of the SAA problems may be driven by a number of individual patients with extreme expected benefits that are either to high or too low. For the process to be able to smoothen this type of outlier data, it should be executed using relatively larger sample sizes. However, the computational limitations due to the size of the pricing problem creates a trade off with respect to solution quality and computational tractability in this aspect as well.

To remedy these computational difficulties, we introduce an aggregate version of the our modeling framework in Chapter 5. This approach aggregates the first J_k patients in each OPO into a single aggregate patient for that particular OPO in every scenario. In other words, the scenarios contain information on what an average patient (who is among the sickest J_k in her OPO) looks like in every OPO, and ranks these average patients in the order of decreasing sickness within the designed regions. Since the number of rows and columns in the pricing problem are driven by the number of patients in each scenario, this approach helps us in scaling the problem down. Hence, it enables us to consider more batches with more scenarios and use bigger region covers.

4.5.1 Evaluating the Results

As mentioned before, it is possible to derive statistical properties and confidence intervals around the quality of solutions obtained using the SAA method. We follow the notation used by Kleywegt and Shapiro [95] during our discussion on the estimation of the optimality gap.

Let v^* denote the *true* optimal objective value of the RMP (4.6). Let $\hat{v}_K^b$ denote the optimal objective vale of the b^{th} SAA replication, i.e., b^{th} batch of K scenarios. Then, an

Table 4.4: Estimating the optimality gap for candidate solutions with $B = 10, K = 200$.

Batch (b)	Map ref.	$\bar{g}_K^B(\hat{x}_K^b)$	$\hat{v}_K^b$	Opt. gap ($\bar{v}_K^B - \bar{g}_K^B(\hat{x}_K^b)$)		95% CI				
						Absolute		% of $\bar{v}_K^B$		
				Mean	Var.	LL	UL	LL	Mean	UL
1	4A	2115.34	2110.40	66.22	94.12	50.27	82.18	2.30%	3.04%	3.77%
2	4B	2095.12	2253.63	86.45	112.42	69.01	103.89	3.16%	3.96%	4.76%
3	4C	2090.98	2120.34	90.59	164.19	69.51	111.67	3.19%	4.15%	5.12%
4	4D	2106.37	2152.31	75.19	88.44	59.72	90.66	2.74%	3.45%	4.16%
5	4E	2095.24	2130.04	86.33	121.90	68.17	104.49	3.12%	3.96%	4.79%
6	4F	2101.52	2232.75	80.05	86.10	64.79	95.31	2.97%	3.67%	4.37%
7	4G	2090.24	2238.70	91.32	142.51	71.69	110.96	3.29%	4.19%	5.09%
8	4H	2100.73	2168.53	80.84	121.39	62.72	98.96	2.87%	3.71%	4.54%
9	4I	2102.66	2169.88	78.91	90.61	63.25	94.56	2.90%	3.62%	4.33%
10	4J	2081.61	2239.08	99.96	202.53	76.55	123.37	3.51%	4.58%	5.66%

estimator of v^* is

$$\bar{v}_K^B = \frac{1}{B}\sum_{b=1}^{B} \hat{v}_K^b. \tag{4.27}$$

Let $g(\hat{x})$ denote the real objective value of candidate solution $\hat{x}$, and $\hat{g}_K^b(\hat{x})$ denote the sample average objective value at $\hat{x}$ of the b^{th} SAA sample of K scenarios. Hence,

$$\hat{g}_K^b(\hat{x}) = \frac{1}{K}\sum_{k\in K^b} \hat{g}_K(\hat{x}), \tag{4.28}$$

where K^b is the set of K scenarios in the b^{th} batch. Let $\bar{g}_K^B(\hat{x})$ denote the mean sample average objective value of solution $\hat{x}$ for B batches, i.e.,

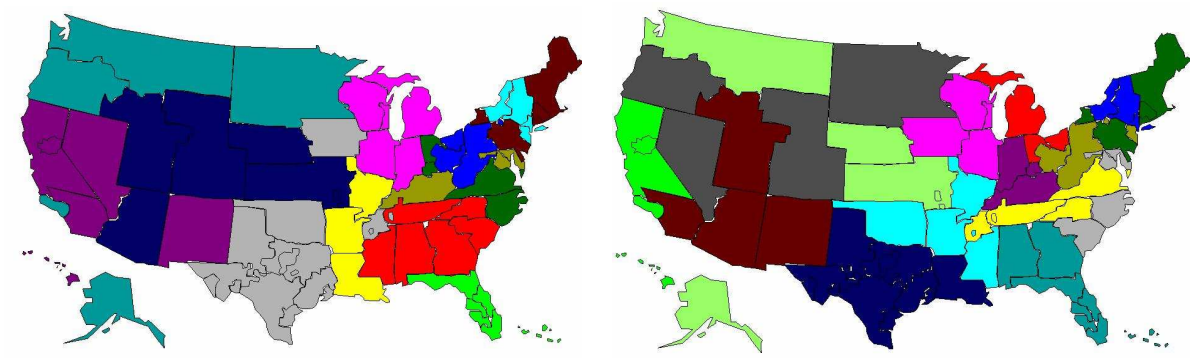
$$\bar{g}_K^B(\hat{x}) = \frac{1}{K}\sum_{b=1}^{B} \hat{g}_K^b(\hat{x}). \tag{4.29}$$

Then, an estimator of the optimality gap of candidate solution $\hat{x}$, i.e., $v^* - g(\hat{x})$, is $\bar{v}_K^B - \bar{g}_K^B(\hat{x})$. And the variance of $\bar{v}_K^B - \bar{g}_K^B(\hat{x})$ is estimated as

$$\frac{\bar{S}^2}{B} = \frac{1}{B(B-1)}\sum_{b=1}^{B}[(\hat{v}_K^b - \hat{g}_K^b(\hat{x})) - (\bar{v}_K^B - \bar{g}_K^B(\hat{x}))]^2. \tag{4.30}$$

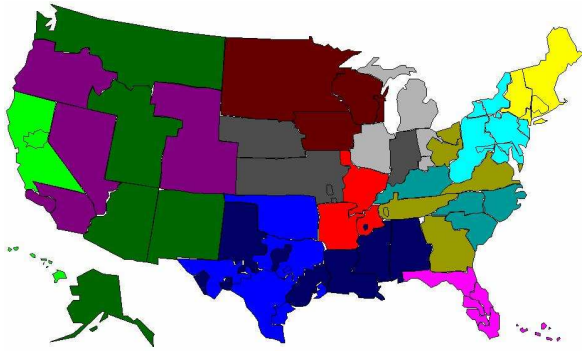
Furthermore, a $(1-\alpha)100\%$ confidence interval around the optimality gap for solution $\hat{x}$ can be computed as

$$\bar{v}_K^B - \bar{g}_K^B(\hat{x}) \pm z_\alpha \frac{\bar{S}}{\sqrt{B}}. \tag{4.31}$$

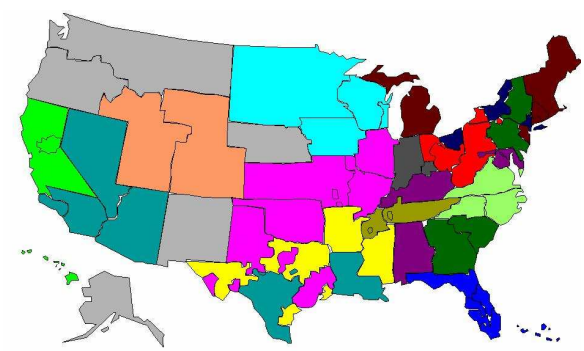


(4A)

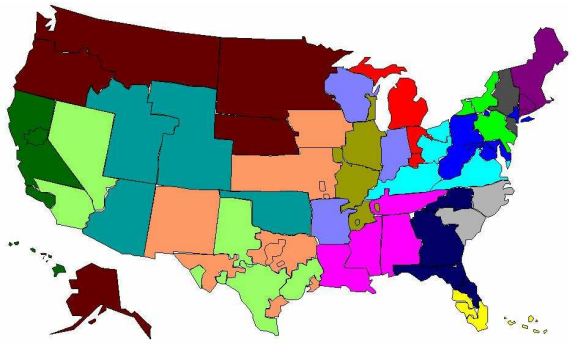
(4B)



(4C)



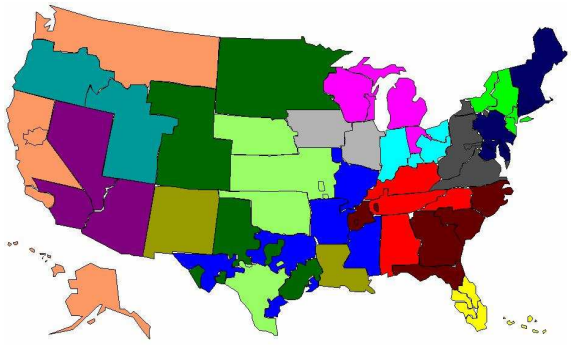
(4D)



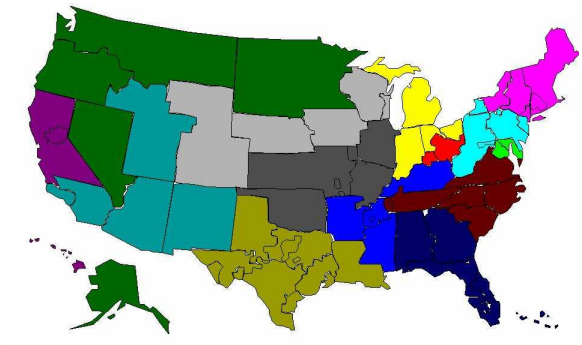
(4E)

Figure 4.2: Maps of regions obtained with geographic decomposition scheme 20_10 with $B = 10, K = 200$.

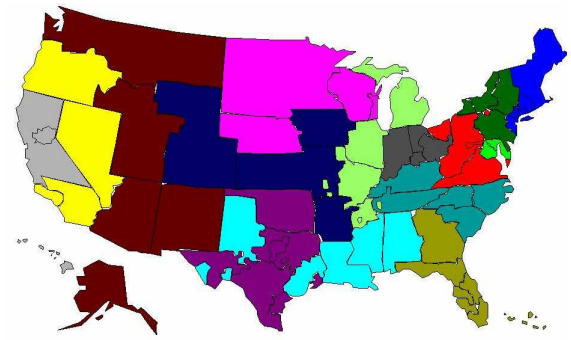
Let $\hat{x}_K^b$ denote the optimal solution of the SAA problem associated with batch b, for $b = 1, \ldots, B$. We evaluated the estimates of optimality gaps for the optimal solutions of our 10 batches of 200 scenarios. Table 4.4 shows the results of the analysis. The *map reference* column shows the letter index that corresponds to the map of each configuration in Figures 4.2 and 4.3, which display the optimal configurations for each batch of scenarios. Note that



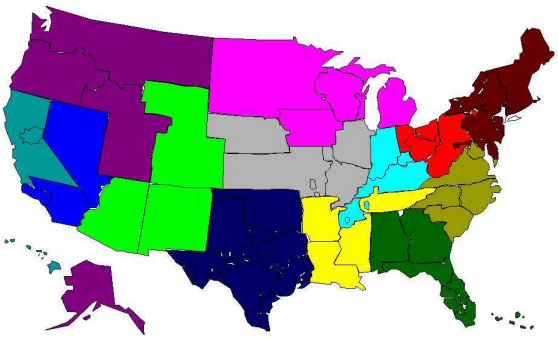
(4F)



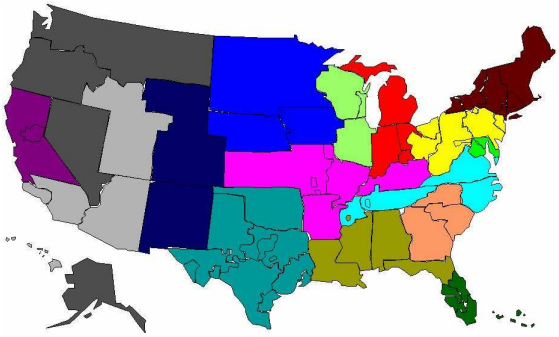
(4G)



(4H)



(4I)



(4J)

Figure 4.3: Maps of regions obtained with geographic decomposition scheme 20_10 with $B = 10, K = 200$ continued.

the optimality gap estimates are valid for the case when the RMP (4.6) is solved under the same geographic decomposition scheme as the SAA problem RMP_{SAA} for every batch, which is 20_10 in our case.

In order to validate the solutions obtained, we simulated the regional configurations shown in Figures 4.2 and 4.3 along with the current regional configuration using the simu-

lation model of Shechter et al. [155]. We simulated the liver allocation system from 1996 to the end of 2002, and took 20 replications of each configuration using a warm-up period of 3 years. In order to compare the performance of our alternative configurations with the current system, we conducted paired t tests. The results of the analysis can be seen in Table 4.5, and Figures 4.4 and 4.5. The null hypothesis of these tests is that the average benefits gained per transplant are equal between an alternative configuration and the current regional configuration. Figure 4.4 displays the 95% confidence intervals around the mean difference obtained in the expected benefit of each configuration when compared to the performance of the current system while Figure 4.5 shows the same results as a percentage of the current system. Since all $p-$values are 0, we conclude that the proposed regional configurations bring significant increases to the expected benefits gained from liver transplants.

Note that Figures 4.2 and 4.3 present a mix of contiguous and noncontiguous regional configurations. The regional configuration currently in use is composed only of contiguous regions. However, contiguity is not listed as a requirement for the design of the liver allocation system. In fact, some OPOs in the organ procurement and transplant network are noncontiguous, as can be seen from Figure 1.2. If needed, it is possible to postprocess noncontiguous solutions using local search techniques to generate contiguous regions.

4.6 CONCLUSIONS

In this chapter, we introduced a stochastic mixed-integer programming model to design liver transplant regions under uncertainty in order to maximize the expected life-time gained by transplants at the regional level. We utilized a set-partitioning master problem with a closed-form expression representing the second-stage objective values in terms of first-stage region selection variables. We solved the problem using a column generation approach that generates favorable regions with respect to different snapshots of the national waiting list composition in the root node of the branch-and-bound tree. We introduced branching and heuristic integer-solution generation routines to take advantage of the special structure of the pricing problems.

Table 4.5: Paired t tests and confidence intervals on the difference in expected life-days gained per transplant: Regional configurations of Figures 4.2 and 4.3 vs. current system

Regional Configuration	Paired Differences					t	$p-$value
	Mean	Standard Deviation	Standard Error	95% CI LL	95% CI UL		
4A	142.29	45.62	10.20	120.94	163.64	13.95	0.0000
4B	116.22	45.11	4.45	106.91	125.53	26.12	0.0000
4C	153.88	47.44	7.53	138.11	169.64	20.43	0.0000
4D	112.42	44.53	4.99	101.98	122.86	22.54	0.0000
4E	100.89	41.18	4.12	92.27	109.51	24.49	0.0000
4F	156.35	47.93	5.09	145.69	167.01	30.70	0.0000
4G	142.59	41.75	3.79	134.66	150.52	37.62	0.0000
4H	89.73	33.79	3.08	83.28	96.18	29.11	0.0000
4I	146.15	47.15	4.77	136.18	156.13	30.67	0.0000
4J	244.72	58.45	5.79	232.60	256.84	42.26	0.0000

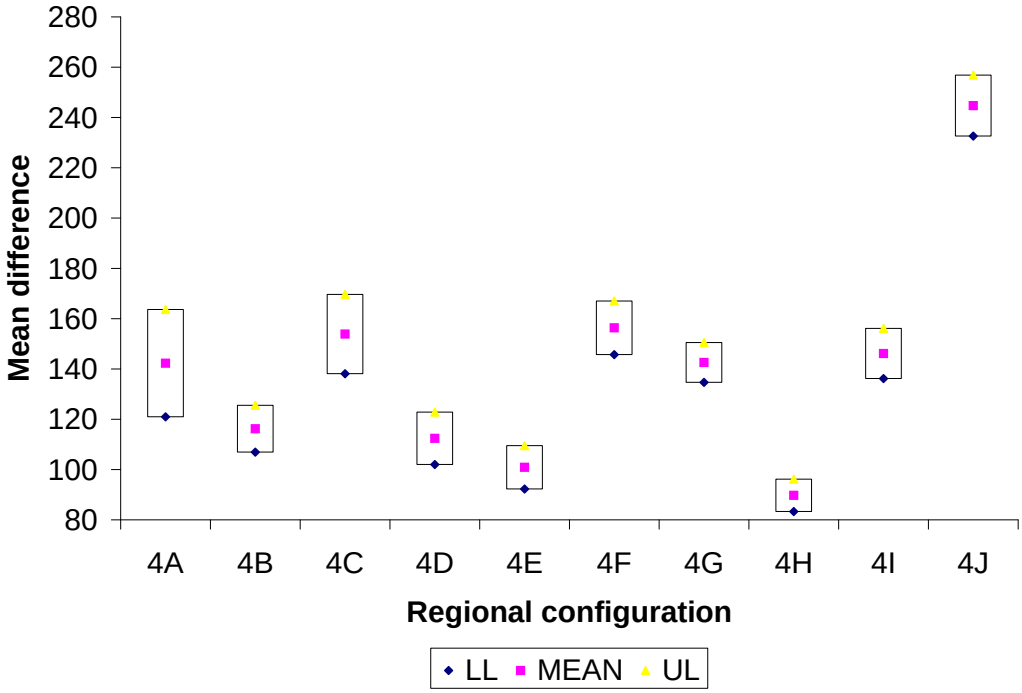


Figure 4.4: 95% confidence intervals around the mean difference in the expected life-days gained per transplant. Detailed results can be seen in Table 4.5.

We utilized a geographic decomposition scheme with 20 region covers with 10 OPOs and generated 10 independent samples of 200 scenarios in our computational experiments. Despite the fact that our solution approaches beat the performance of using CPLEX with default settings by an average of 35% with respect to the solution time, we observed that the size of the large-scale pricing problems gives us limited flexibility in being able to solve the problem using larger region covers, more samples or bigger sample sizes. Hence, we introduce an aggregate version of the modeling framework in Chapter 5 that enables us to perform more extensive testing.

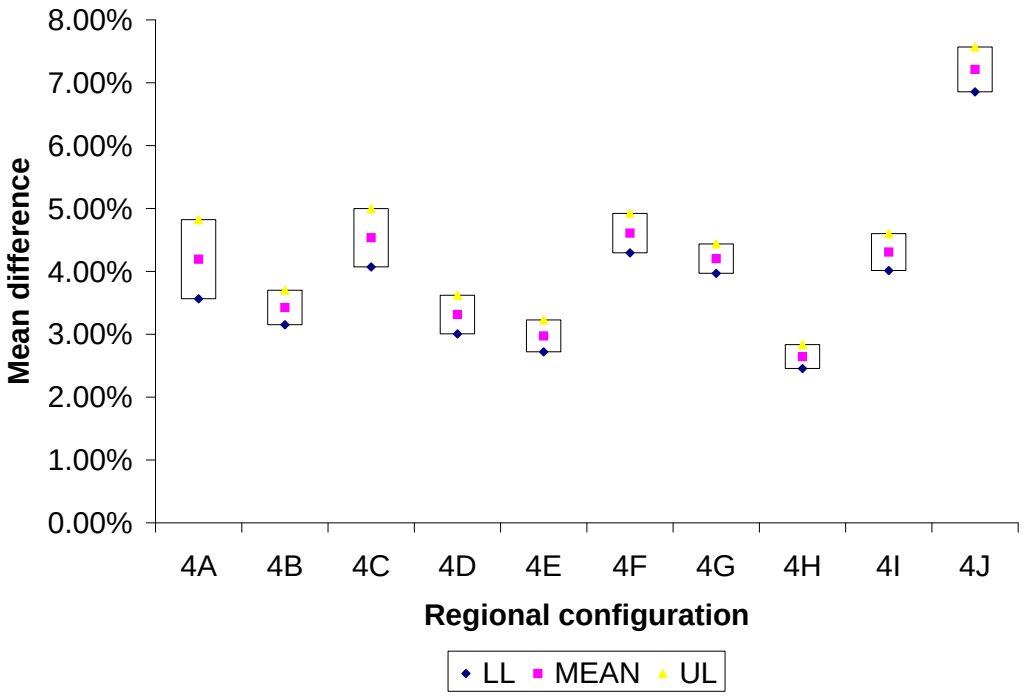


Figure 4.5: 95% confidence intervals around the mean difference in the expected life-days gained per transplant as percentage of the performance of the current system. Detailed results can be seen in Table 4.5.

Despite computational limitations, our results appear to be encouraging with respect to estimated optimality gaps and the validity analysis we performed using the liver allocation system simulation model of Shechter et al. [155]. The 10 alternative configurations presented in this chapter have an average optimality gap of around 4% and the statistical tests on simulation output suggests that an increase of around 5% in expected life-time gained with a transplant can be attained using these region designs. Particularly, regional configuration 4J of Figure 4.3 appears to posses the potential of increasing the expected life-time gained by 7% when compared to the performance levels of the system today.

5.0 DESIGNING LIVER TRANSPLANT REGIONS UNDER UNCERTAINTY USING AN OPO-BASED MODEL

5.1 INTRODUCTION

As we discussed in Chapter 4, solving a model that considers every individual patient is challenging. The model introduced in Chapter 4 does not lend itself well to computational tests using larger samples of scenarios or geographic decomposition schemes with bigger region covers. Moreover, as the number of scenarios that can be considered while solving the model will be limited, the solutions will tend be sensitive to outlier patients.

In this chapter, we introduce an alternative aggregate model for the problem of designing liver transplant regions under uncertainty. We keep most of the modeling framework of Chapter 4 intact, however, aggregate the patients in every scenario into a single aggregated patient per OPO whose characteristics reflect the average over the patients in the OPO. Hence, the aggregate model relies on an aggregate set of patients in each scenario. This approach scales down the size of the pricing problem and enables the use of larger samples of scenarios and bigger geographic decomposition schemes. Intuitively, it is also less prone to outlier patients with extreme characteristics since it averages patients out at the OPO level.

The aggregate model we develop in this chapter and the one introduced in Chapter 4 present a trade off in terms of modeling the real-world system and computational tractability. While Chapter 4 focuses on a more accurate representation of liver offers by considering the probabilities of individual patient receiving offers, it develops a model that is very hard to solve. The aggregate model, on the other hand, promises better computational performance at the expenses of losing some level of detail through the aggregation of patients.

Once again we employ a column generation approach in which we perform extensive pricing in the root node of the branch-and-bound tree. We solve a set of Sample Average Approximation problems using different samples and geographic decomposition schemes. We perform validation analysis on the results using statistical results from the SAA literature and the liver allocation system simulation of Shechter et al. [155].

This chapter is organized as follows: in Section 5.2 we discuss the aggregate modeling framework and how it differs from the model introduced in Chapter 4. In Section 5.3, we present the computational approaches we utilize to solve and analyze the problem. Section 5.4 contains discussions on our computational results and validation analysis. And finally, in Section 5.5, we summarize our conclusions for this chapter.

5.2 AN ALTERNATIVE AGGREGATE MODEL FOR REGION DESIGN UNDER UNCERTAINTY

We extend the notation developed in Chapter 4. The aggregate model is a modification of the stochastic programming model introduced in Chapter 4 where the patient set in a scenario ξ^k, for $k = 1, \ldots, K$, is aggregated so that the set only contains one *average* patient per OPO. We also formulate a new pricing problem, similar to (4.21), for the aggregate model. First, we start by introducing the modification of the patient sets, and hence, the second-stage models.

An *aggregate* patient in OPO $i \in I$ is a hypothetical patient who is created by averaging the problem data for the sickest J_k patients listed in OPO i. Let $\tilde{M}_k$ denote the *set of aggregate patients* under scenario ξ^k, for $k = 1, \ldots, K$. In the aggregate modeling framework, for every scenario $\xi^k \in \Xi$, the set of individual patients M_k is transformed into, and replaced by, the set of aggregate patients, $\tilde{M}_k$. Note that any notation that applies to members of set M_k also applies to $\tilde{M}_k$.

Let $\mu_m(\xi^k)$ denote the MELD score of an individual patient $m \in M_k$, under scenario $\xi^k \in \Xi$. Let ϕ_m denote the probability that patient m gets the organ offer given the liver is offered only to OPO $O(m, k)$. Note that ϕ_m also depends on the scenario, but we drop

index k for ease of exposition. The parameter ϕ_m forces preference among patients listed in the same OPO. For instance, one might want to assign probabilities increasing with respect to the severity of the illness, i.e., $\phi_{m'} > \phi_{m''}$ for $m', m'' \in M_k, O(m') = O(m''), m' < m''$. For every $i \in I$, note that $\sum_{m \in M_k : O(m,k)=i} \phi_m = 1$ for $k = 1, \ldots, K$. Hence, the mean MELD score and expected benefit from the available liver in scenario $\xi^k \in \Xi$ of the top J_k patients in OPO i can be calculated as

$$\mu^i(\xi^k) := \sum_{m \in M_k : O(m,k)=i} \phi_m \mu_m(\xi^k), \tag{5.1}$$

and

$$c^i(\xi^k) := \sum_{m \in M_k : O(m,k)=i} \phi_m c_m(\xi^k), \tag{5.2}$$

respectively.

After calculating these values for every OPO $i \in I$, the aggregate patients are sorted in descending order of $\mu^i(\xi^k)$, i.e., MELD score, and inserted into the set $\tilde{M}_k$ in that order. Hence, as we did for M_k, we assume that $\tilde{M}_k$ is ordered such that if aggregate patient m' is sicker than aggregate patient m'', i.e., has a higher MELD score, then $m' < m''$. Since we assume the available liver is at the regional matching phase, there is no aggregate patient for the donor OPO in scenario ξ^k. Thus, $\{m \in \tilde{M}_k | O(m,k) = O(k)\} = \emptyset$.

Figure 5.1 shows an example where we form the set $\tilde{M}_k$ from M_k for a scenario ξ^k where $J_k = 2, |M_k| = 6$, $\phi_m = \frac{1}{J_k}, \forall m \in M_k$ and $|I| = 3$.

We keep the underlying modeling assumptions of Chapter 4 intact. Let $\tilde{q}_k$ denote the probability that an aggregate patient will accept the organ available under scenario $\xi^k \in \Xi$. We calculate $\tilde{q}_k$ using q_k, i.e., the probability that an individual patient accepts an organ offer, and define it as $Prob\{$Liver in scenario ξ^k is accepted by one of the top J_k patients in an OPO $|$ it is offered to the OPO$\}$. Thus, we have

$$\tilde{q}_k = \sum_{j=1}^{J_k} (1 - q_k)^{j-1} \cdot q_k. \tag{5.3}$$

We modify Assumption 4.3 as follows:

Assumption 5.1. *For every scenario $\xi^k \in \Xi$, the rank of the aggregate patient that will receive the transplantation has a geometric distribution with parameter $\tilde{q}_k$.*

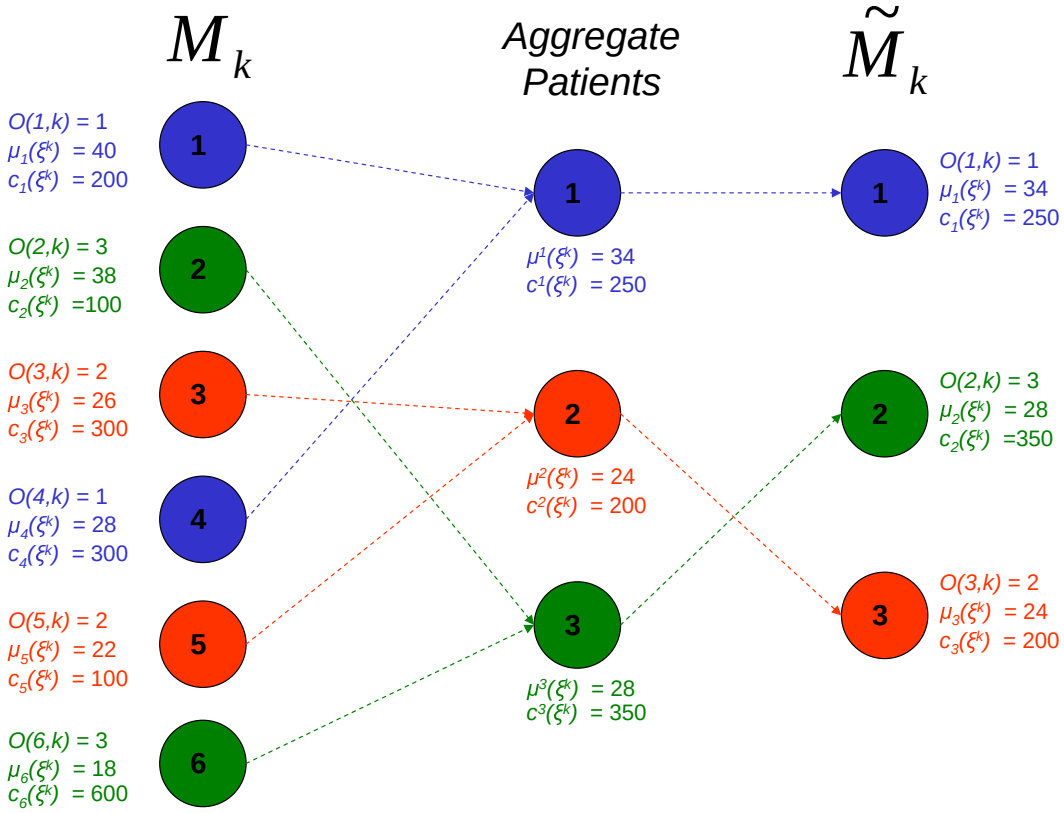


Figure 5.1: An example for creating the set of aggregate patients in a scenario $\xi^k \in \Xi$, where $|I| = 3$, $J_k = 2$, $|M_k| = 6$ and $\phi_m = \frac{1}{J_k}, \forall m \in M_k$. Aggregate patients are formed by averaging problem data for individual patients, and then sorted to form the set $\tilde{M}_k$.

We also utilize the following remark:

Remark 5.1. *For every scenario $\xi^k \in \Xi$, there is exactly one aggregate patient in every OPO $i \in I \backslash \{O(k)\}$.*

If for some scenario $\xi^k \in \Xi$, there is no patient in OPO $i \in I$, we can create a dummy aggregate patient in OPO i with a MELD score of 0 (which guarantees that the dummy aggregate patient does not affect the ranking since the MELD score changes between 6 and 40) and an expected benefit of 0 days. By Remark 5.1, $|\tilde{M}_k| = |I| - 1$, for $k = 1, \ldots, K$.

5.2.1 Column Generation Framework for the Aggregate Model

As in Chapter 4, we utilize a column generation framework in which we generate regions with favorable reduced costs using a pricing problem. The restricted master problem (RMP) for the aggregate model is the same as in Chapter 4, and its formulation is given by (4.6).

We modify the pricing problem introduced in Chapter 4 and use the notation developed in Section 4.2.3. As in the second-stage models, the set of individual patients, M_k, in scenario ξ^k is replaced by the set of aggregate patients, $\tilde{M}_k$, for $k = 1, \ldots, K$, while formulating the pricing problem for the aggregate modeling framework.

The reduced cost of a region $\hat{r} \in R$ can be calculated as follows:

$$\bar{c}_{\hat{r}} = \sum_{k=1}^{K} p^k \left(\sum_{m \in \tilde{M}_k} c_m(\xi^k)\ T_{m\hat{r}}(\xi^k) \right) - \sum_{i \in I} \pi_i a_{i\hat{r}}, \tag{5.4a}$$

$$= \sum_{k=1}^{K} \sum_{m \in \tilde{M}_k} \left(p^k c_m(\xi^k)\ Prob\{m|\hat{r}, k\} \right) - \sum_{i \in I} \pi_i a_{i\hat{r}}. \tag{5.4b}$$

First, recall from Section 4.2.3 that y_i is a binary variable such that $y_i = 1$ if OPO $i \in I$ is chosen for the new region, and $y_i = 0$ otherwise. We modify the definition of variables z_{mjk} such that $z_{mjk} = 1$ if aggregate patient $m \in \tilde{M}_k$ is in j^{th} place in the new region under scenario $\xi^k \in \Xi$, and $z_{mjk} = 0$ otherwise, for $j = 1, \ldots, |I| - 1$.

In addition to the notation of Section 4.2.3, we introduce a set of dummy rank variables. Since $\{m \in \tilde{M}_k | O(m,k) = O(k)\} = \emptyset$ and there can only be one aggregate patient per OPO, the number of spots that have to be filled by aggregate patients is $|I| - 1$ for the aggregate problem. However, in a scenario $\xi^k \in \Xi$, aggregate patients can only be ranked if the donor OPO $O(k)$ and their OPOs are selected to be in the newly designed region. Hence, we need dummy rank variables to make sure every spot is filled. Let D_j be the j^{th} dummy rank variable such that $D_j = 1$ if rank j is not used and $D_j = 0$ otherwise, for $j = 1, \ldots, |I| - 1$. Note that D_j does not depend on scenarios. This is due to the fact that in every scenario $\xi^k \in \Xi$ where $y_{O(m,k)} = 1$, we need exactly $|I| - 1 - \sum_{i \in I \setminus O(m,k)} y_i$ dummy rank variables to be set to 1, and the vector y is independent of the scenarios. Since $y_{O(m,k)} = 1$, we have

$\sum_{i\in I} y_i + \sum_{j=1}^{|I|-1} D_j = |I|$, which is scenario independent. However, we also need to make sure that if $D_{j'} = 1$, then $D_j = 1$ for $j' < j \leq |I| - 1$. Hence, we have the following constraints:

$$\sum_{i\in I} y_i + \sum_{j=1}^{|I|-1} D_j = |I|, \tag{5.5a}$$

$$D_j \leq D_{j+1}, \qquad j = 1, \ldots, |I| - 2. \tag{5.5b}$$

As mentioned before, for every scenario ξ^k, an aggregate patient $m \in \tilde{M}_K$ can be ranked in one of the spots 1 through $|I| - 1$, if (a) her OPO is chosen for the new region, i.e., $y_{O(m,k)} = 1$, and (b) the donor OPO for scenario ξ^k is in the new region, i.e. $y_{O(k)} = 1$. Thus,

$$\sum_{j=1}^{m} z_{mjk} \leq y_{O(m,k)}, \qquad k = 1, \ldots, K, \forall m \in \tilde{M}_k, \tag{5.6a}$$

$$\sum_{j=1}^{m} z_{mjk} \leq y_{O(k)}, \qquad k = 1, \ldots, K, \forall m \in \tilde{M}_k. \tag{5.6b}$$

The following set of constraints enforces the ranks to be assigned in decreasing order of illness severity within the region:

$$\sum_{j=1}^{m} j \cdot z_{mjk} + m \cdot (2 - y_{O(k)} - y_{O(m,k)}) \geq \sum_{i\in I} n^k_{mi} \cdot y_i, \qquad k = 1, \ldots, K, \forall m \in \tilde{M}_k. \tag{5.7}$$

Note that (5.7) is automatically satisfied for scenario $\xi^k \in \Xi$ and patient $m \in \tilde{M}_k$ if either $y_{O(k)} = 0$, or $y_{O(m,k)} = 0$ (or both). However, if $y_{O(k)} = y_{O(m,k)} = 1$, it enforces the rankings to follow the order of decreasing sickness levels.

Each of the spots 1 through $|I| - 1$, for scenario ξ^k, have to be filled if the donor OPO for that particular scenario is a part of the new region. Moreover, a spot can be assigned to at most one aggregate patient or dummy rank variable. Hence, we have

$$\sum_{m=j}^{|I|-1} z_{mjk} + D_j \geq y_{O(k)}, \qquad k = 1, \ldots, K, j = 1, \ldots, |I| - 1, \tag{5.8a}$$

$$\sum_{m=j}^{|I|-1} z_{mjk} + D_j \leq 1, \qquad k = 1, \ldots, K, j = 1, \ldots, |I| - 1. \tag{5.8b}$$

Since we only consider organ-patient matching at the regional phase, for the benefits of a scenario to be realized, the liver available under that scenario has to be in the selected region, and we can only evaluate the benefits of transplant for aggregate patients in that particular region.

The system of inequalities described by (5.5)-(5.8) forms the constraint set of our pricing problem. A very important observation is that for fixed binary vectors $y \in \mathbb{B}^{|I|}$ and $D \in \mathbb{B}^{|I|-1}$, this system is separable for each scenario $k = 1, \ldots, K$. Table 5.1 shows the structure of the constraint matrix formed by the system (5.5)-(5.8). Each row in the table corresponds to the coefficient matrix for an individual scenario, except for the first row which corresponds to the first-stage constraint set (5.5), and the thick dots represent variable groups that have nonzero coefficients for the constraint sets of that particular scenario. From Table 5.1, it is easy to see that once the y and D variables are fixed, we are left with K independent subsystems, each of which corresponds to an individual scenario in Ξ.

For a fixed OPO vector $y \in \mathbb{B}^{|I|}$, dummy rank vector $D \in \mathbb{B}^{|I|-1}$ such that (y, D) satisfies (5.5), let $P(y, D, k) = \{z | (5.9a) - (5.9f)\}$ for scenario $\xi^k \in \Xi$, where

$$\sum_{j=1}^{m} z_{mj} \leq y_{O(m,k)}, \qquad \forall m \in \tilde{M}_k, \tag{5.9a}$$

$$\sum_{j=1}^{m} z_{mj} \leq y_{O(k)}, \qquad \forall m \in \tilde{M}_k, \tag{5.9b}$$

$$\sum_{j=1}^{m} j \cdot z_{mj} + m \cdot (2 - y_{O(k)} - y_{O(m,k)}) \geq \sum_{i \in I} n^k_{mi} \cdot y_i, \qquad \forall m \in \tilde{M}_k, \tag{5.9c}$$

$$\sum_{m=j}^{|I|-1} z_{mj} + D_j \geq y_{O(k)}, \qquad j = 1, \ldots, |I| - 1, \tag{5.9d}$$

$$\sum_{m=j}^{|I|-1} z_{mj} + D_j \leq 1, \qquad j = 1, \ldots, |I| - 1. \tag{5.9e}$$

$$0 \leq z_{mj} \leq 1, \qquad \forall m \in \tilde{M}_k, j = 1, \ldots, m. \tag{5.9f}$$

Now, we will prove that a solution to the system $P(y, D, k)$ ranks the aggregate patients under scenario ξ^k in decreasing order of sickness in the region defined by y. In order to prove this claim, we need the following set of lemmas. Lemmas 5.1 through 5.3 mirror Lemmas 4.1 through 4.3 and therefore the proofs are omitted.

Table 5.1: Constraint structure of the system of inequalities represented by (5.5)-(5.8).

Scenario	Variables					
	y	D	z_{mj1}	z_{mj2}	$\cdots$	z_{mjK}
-	•	•				
		•				
1	•		•			
	•		•			
	•		•			
	•	•	•			
		•	•			
2	•			•		
	•			•		
	•			•		
	•	•		•		
		•		•		
	$\vdots$				$\ddots$	
K	•					•
	•					•
	•					•
	•	•				•
		•				•

Lemma 5.1. *For a fixed $y \in \mathbb{B}^{|I|}$ and scenario $\xi^k \in \Xi$,*

(a) if $y_{O(k)} = 0$, then $z_{mj} = 0$, $\forall m \in \tilde{M}_k$, $j = 1, \ldots, m$,

(b) if $y_{O(k)} = 1$, then for a patient $m \in \tilde{M}_k$ such that $y_{O(m,k)} = 0$, $z_{mj} = 0$, for $j = 1, \ldots, m$.

Lemma 5.2. *For a scenario $\xi^k \in \Xi$, patient $m \in \tilde{M}_k$ and an integer $1 \leq \ell \leq m$*

$$\left\{ \max \sum_{j=1}^{\ell} j \cdot z_{mj} | \sum_{j=1}^{\ell} z_{mj} \leq 1, z_{mj} \geq 0, j = 0, \ldots, \ell \right\} = \ell.$$

Lemma 5.3. *For an OPO vector $y \in \mathbb{B}^{|I|}$, scenario $\xi^k \in \Xi$ and patient $m \in \tilde{M}_k$, $m \geq \sum_{i \in I} n^k_{mi} y_i$.*

We also need to modify some notation. For $y \in \mathbb{B}^{|I|}$ and $k = 1, \ldots, K$, let $\tilde{m}(y, \ell)$ denote the index $m' \in \tilde{M}_k$ such that $\sum_{i \in I} n^k_{m'i} y_i = \ell$ where $\tilde{M}(y, k) = \{m \in \tilde{M}_k | y_{O(m,k)} = 1\}$, i.e., the set of aggregate patients in the region defined by y. Note that $\tilde{m}(y, \ell)$ shows the index of the aggregate patient whose regional rank is ℓ under scenario ξ^k. Although $\tilde{m}(y, \ell)$ also depends on scenario ξ^k, we drop k from the notation for ease of exposition.

Lemma 5.4. *Consider a fixed OPO vector $y \in \mathbb{B}^{|I|}$ and dummy rank vector $D \in \mathbb{B}^{|I|-1}$ such that (y, D) satisfies (5.5). Suppose $y_{O(k)} = 1$ for scenario $\xi^k \in \Xi$. Then, for $|\tilde{M}(y, k)| < \ell \leq |I| - 1$, $D_\ell = 1$.*

Proof. Since $y_{O(k)} = 1$, (5.9d) and (5.9e) together imply

$$\sum_{m=j}^{|I|-1} z_{mj} + D_j = 1, \qquad j = 1, \ldots, |I| - 1. \tag{5.10}$$

Summing up (5.10) over the spots that have to filled, i.e., $j = 1, \ldots, |I| - 1$, yields

$$\sum_{j=1}^{|I|-1} \left(\sum_{m=j}^{|I|-1} z_{mj} + D_j \right) = \sum_{j=1}^{|I|-1} D_j + \sum_{j=1}^{|I|-1} \sum_{m=j}^{|I|-1} z_{mj} = |I| - 1. \tag{5.11}$$

By the definition of $\tilde{M}(y, k)$, summing up (5.9a) over $m \in \tilde{M}_k$ gives

$$\sum_{m=1}^{|I|-1} \sum_{j=1}^{m} z_{mj} \leq \sum_{m=1}^{|I|-1} y_{O(m,k)} = |\tilde{M}(y, k)|. \tag{5.12}$$

By changing the summation terms in (5.12) we get

$$\sum_{m=1}^{|I|-1}\sum_{j=1}^{m} z_{mj} = \sum_{j=1}^{|I|-1}\sum_{m=j}^{|I|-1} z_{mj} \leq |\tilde{M}(y,k)|. \tag{5.13}$$

Hence, by (5.11) and (5.13), we have

$$\sum_{j=1}^{|I|-1} D_j \geq |I| - 1 - |\tilde{M}(y,k)|. \tag{5.14}$$

Finally, combining (5.14) with (5.5b) and the fact that $D_j \in \{0,1\}$, for $j = 1,\ldots,|I|-1$, gives

$$D_j = 1, \text{ for } j = |\tilde{M}(y,k)|+1,\ldots,|I|-1.$$

□

Lemma 5.5. *Consider a fixed OPO vector $y \in \mathbb{B}^{|I|}$ and dummy rank vector $D \in \mathbb{B}^{|I|-1}$ such that (y,D) satisfies (5.5). Suppose $y_{O(k)} = 1$ for scenario ξ^k. Then, for $\ell = 1,\ldots,|\tilde{M}(y,k)|$, (a) $D_\ell = 0$, and (b) $z_{\tilde{m}(y,\ell)\ell} = 1$, and $z_{\tilde{m}(y,\ell)j} = 0$ for $j = 1,\ldots,|I|-1, j \neq \ell$.*

Proof. Since $y_{O(k)=1}$ by assumption, and $y_{O(m,k)} = 1$ for $\ell = 1,\ldots,|\tilde{M}(y,k)|$ by the definition of $\tilde{M}(y,k)$, (5.9c) reduces to

$$\sum_{j=1}^{\tilde{m}(y,\ell)} j \cdot z_{\tilde{m}(y,\ell)j} \geq \ell, \qquad\qquad \ell = 1,\ldots,|\tilde{M}(y,k)|. \tag{5.15}$$

By Lemma 5.4, $D_j = 1$ for $|\tilde{M}(y,k)| < j \leq |I|-1$. Hence, by (5.10), $z_{\tilde{m}(y,\ell)j} = 0$ for $\ell = 1,\ldots,|\tilde{M}(y,k)|$ and $j = |\tilde{M}(y,k)|+1,\ldots,|I|-1$. Thus, since $\tilde{m}(y,\ell) \geq \ell$ by Lemma 5.3, for $\ell = |\tilde{M}(y,k)|$, (5.15) becomes

$$\sum_{j=1}^{|\tilde{M}(y,k)|} j \cdot z_{\tilde{m}(y,|\tilde{M}(y,k)|)j} \geq |\tilde{M}(y,k)|. \tag{5.16}$$

By (5.16) and Lemma 5.2, $z_{\tilde{m}(y,|\tilde{M}(y,k)|)|\tilde{M}(y,k)|} = 1$, and by (5.9a), $z_{\tilde{m}(y,|\tilde{M}(y,k)|)j} = 0$ for $j = 1,\ldots,|\tilde{M}(y,k)|-1$. Furthermore, by (5.10), $z_{\tilde{m}(y,\ell)|\tilde{M}(y,k)|} = 0$ for $\ell = 1,\ldots,|\tilde{M}(y,k)|$, and $D_{|\tilde{M}(y,k)|} = 0$. Hence, by (5.5b), $D_j = 0$ for $= 1,\ldots,|\tilde{M}(y,k)|$, which proves proves part *(a)*.

Assume the claim for part *(b)* holds for $\ell = l, \ldots, |\tilde{M}(y,k)|$. We have already shown that it holds for $\ell = |\tilde{M}(y,k)|$. Then, for $\ell = l-1$, constraint (5.9c) reduces to

$$\sum_{j=1}^{l-1} j \cdot z_{\tilde{m}(y,l-1)j} \geq l-1. \tag{5.17}$$

By Lemma 5.2 constraint (5.17) is only satisfied if $z_{\tilde{m}(y,l-1)l-1} = 1$. From (5.9a) and (5.10), it follows that $z_{\tilde{m}(y,l-1)j} = 0$ for $j = 1, \ldots, |I|-1, j \neq l-1$, and $z_{\tilde{m}(y,\ell)l-1} = 0$ for $\ell = 1, \ldots, |\tilde{M}(y,k)|, \ell \neq l-1$. Hence, by induction, part *(b)* is proven. □

We are now ready to state Theorem 5.1. The proof follows from Lemmas 5.1 through 5.5.

Theorem 5.1. *For a fixed OPO vector $y \in \mathbb{B}^{|I|}$ and dummy rank vector $D \in \mathbb{B}^{|I|-1}$ such that (y, D) satisfies (5.5), a solution to the system $P(y, D, k)$ ranks the aggregate patients under scenario $\xi^k \in \Xi$ in decreasing order of sickness in the region defined by y.*

We have already discussed that the scenario constraint matrices are separable once fixed y and D are given (see Table 5.1). Hence, the following corollary follows:

Corollary 5.1. *For a fixed OPO vector $y \in \mathbb{B}^{|I|}$ and dummy rank vector $D \in \mathbb{B}^{|I|-1}$ such that (y, D) satisfies (5.5), a solution to the system*

$$P(y,D) = \{0 \leq z_{mjk} \leq 1 \text{ for } k = 1, \ldots, K, m \in \tilde{M}_k, j = 1, \ldots, m | (5.6) - (5.8)\},$$

ranks the aggregate patients under all scenarios in Ξ in decreasing order of sickness in the region defined by y.

We modify $\tilde{c}_{mjk}$ of Section 4.2.3 as follows:

$$\tilde{c}_{mjk} = \begin{cases} (1-\tilde{q}_k)^{j-1} \cdot \tilde{q}_k \cdot c_m(\xi^k), & \text{if } 1 \leq j \leq m, \\ 0, & \text{otherwise,} \end{cases} \quad \text{for } k = 1, \ldots, K, m \in \tilde{M}_k. \tag{5.18}$$

The parameter $\tilde{c}_{mjk}$ can be interpreted as the contribution of aggregate patient m to the expected benefit of liver transplantation for scenario ξ^k if the patient is ranked j^{th}.

The pricing problem needs to be able to handle the ranking of aggregate patients for different regions for every scenario, and needs to incorporate the patient-acceptance probabilities that are dependent on these rankings. The following formulation manages this:

$$SRPP(\pi, I, K) = \max \sum_{k=1}^{K} \sum_{m \in \tilde{M}_k} \sum_{j=1}^{m} p^k \cdot \tilde{c}_{mjk} \cdot z_{mjk} - \sum_{i \in I} \pi_i \cdot y_i \tag{5.19a}$$

$$\text{subject to} \quad \sum_{i \in I} y_i + \sum_{j=1}^{|I|-1} D_j = |I|, \tag{5.19b}$$

$$D_j \leq D_{j+1}, \qquad j = 1, \ldots, |I| - 2, \tag{5.19c}$$

$$\sum_{j=1}^{m} z_{mjk} \leq y_{O(m,k)}, \qquad k = 1, \ldots, K, \forall m \in \tilde{M}_k, \tag{5.19d}$$

$$\sum_{j=1}^{m} z_{mjk} \leq y_{O(k)}, \qquad k = 1, \ldots, K, \forall m \in \tilde{M}_k, \tag{5.19e}$$

$$\sum_{j=1}^{m} j \cdot z_{mjk} + m \cdot (2 - y_{O(k)} - y_{O(m,k)}) \geq \sum_{i \in I} n^k_{mi} \cdot y_i, \quad k = 1, \ldots, K, \forall m \in \tilde{M}_k, \tag{5.19f}$$

$$\sum_{m=j}^{|I|-1} z_{mjk} + D_j \geq y_{O(k)}, \qquad k = 1, \ldots, K, j = 1, \ldots, |I| - 1, \tag{5.19g}$$

$$\sum_{m=j}^{|I|-1} z_{mjk} + D_j \leq 1, \qquad k = 1, \ldots, K, j = 1, \ldots, |I| - 1, \tag{5.19h}$$

$$y_i \in \{0, 1\}, \qquad \forall i \in I, \tag{5.19i}$$

$$D_j \in \{0, 1\}, \qquad j = 1, \ldots, |I| - 1, \tag{5.19j}$$

$$0 \leq z_{mjk} \leq 1, \qquad k = 1, \ldots, K, \forall m \in \tilde{M}_k, j = 1, \ldots, m. \tag{5.19k}$$

The objective function (5.19a) maximizes the reduced cost of a region design. Constraint (5.19b) makes sure that the number of OPOs chosen in a region plus the number of dummy ranks used is equal to the total number of OPOs. Constraint set (5.19c) ensures that if the dummy rank variable for spot j' is used, dummy rank variables for spots $j' < j \leq |I| - 1$

also have to be used. Constraint set (5.19d) makes sure that patients can only be ranked if they are listed in an OPO that is chosen for new region design. Similarly, constraint (5.19e) makes sure that patients are ranked only for scenarios that apply to the new region, i.e., scenarios under which a liver is harvested within the region. Constraint (5.19f) enforces the aggregate patients to be ranked in descending order of the severity of illness. Constraint set (5.19g) forces every spot from 1 to $|I|-1$ to be assigned either an aggregate patient or dummy rank variable if scenario ξ^k has a harvested organ within the new region, while (5.19h) limits the number of aggregate patients or dummy rank variables assigned to a spot to at most 1. And finally, constraints (5.19i) and (5.19j) force variables y_i and D_j to take on binary values, while constraint (5.19k) determines the range for variables z_{mjk}.

Note that the pricing problem (5.19) is a two-stage stochastic mixed-integer program and its deterministic equivalent can be written as follows:

$$SRPP(\pi, I, K) = \max \mathbb{E}_\xi \tilde{Q}(y, D, \xi^k) - \sum_{i \in I} \pi_i \cdot y_i \tag{5.20a}$$

$$\text{subject to} \quad (5.19b) - (5.19c),$$

$$y_i \in \{0,1\}, \qquad \forall i \in I, \tag{5.20b}$$

$$D_j \in \{0,1\}, \qquad j = 1, \ldots, |I|-1, \tag{5.20c}$$

where

$$\tilde{Q}(y, D, \xi^k) = \left\{ \max \sum_{m \in \tilde{M}_k} \sum_{j=1}^{m} \tilde{c}_{mj} z_{mj} | (5.9a) - (5.9f) \right\}, \quad \text{for } k = 1, \ldots, K. \tag{5.21}$$

Suppose we solve this pricing problem to optimality, and obtain the optimal solution vector (y^*, D^*, z^*). Then, y^* gives the region design with maximum reduced cost. Let us denote this region by $\hat{r}$. If the objective value (5.19a) is strictly positive, we add $\hat{r}$ to the restricted set of regions, R', and the column that corresponds to region $\hat{r}$ in formulation (4.6) can be constructed as follows:

- Objective function (4.6a) coefficient for region $\hat{r}$: $d_{\hat{r}} = \sum_{k=1}^{K} p^k \left(\sum_{m \in M_k} \tilde{c}_{mjk} \cdot z^*_{mjk} \right)$,
- For constraint set (4.6b), $a_{i\hat{r}} = y^*_i, \quad \forall i \in I$.

5.3 COMPUTATIONAL APPROACHES AND PARAMETER ESTIMATION

Since the aggregate modeling framework introduced in this chapter is a modification of the model in Chapter 4, similar concerns regarding the scenario space and the size of the pricing problem, which we discussed in Section 4.3, also apply to the aggregate model. We address solution methods that address these computational concerns in this section. We also discuss our parameter estimation procedures.

As there are too many scenarios in the real-world system to consider explicitly from a modeling and solution perspective, once again, we utilize the *Sample Average Approximation* (SAA) method. As we discussed in Section 4.3.1, under the SAA method, the RMP (4.6) is replaced with RMP_{SAA}, i.e.,(4.25), and RMP_{SAA} is solved B times using B independent samples of K scenarios. See Section 4.3.1 for more details.

We also solve RMP_{SAA} using SPRINT methods from airline crew scheduling literature [12], which involve extensive pricing *only* in the root node of a branch-and-bound tree to aid us in the solution process. See Section 4.3.2 for more details.

The pricing problem (5.19) introduced in this chapter constitutes a large scale stochastic mixed-integer program that has to be solved repeatedly throughout the course of the SPRINT algorithm. The number of rows in the pricing problem is $O(K \cdot |I|)$ and the number of columns is $O(K \cdot |I|^2)$, out of which $O(|I|)$ are binaries. In order to reduce the size of the pricing problem we employ the *Geographic Decomposition* approach of Kong [96] and Kong et al. [97], as we did in Chapters 3 and 4. Thus, instead of considering a very large pricing problem for the whole nation, we create a number of smaller pricing problems, each covering different, but overlapping, geographic areas of the country, called *region covers*. See Section 4.3.3 for the application of geographic decomposition within our proposed modeling framework, and see Section 3.3.1.1 for further details on this technique.

We also execute the branching strategy of Ryan and Foster [148] which has proven to be effective on set-partitioning type problems. See Section 3.3.1.2 for further details on this branching strategy.

5.3.1 Data Sources and Parameter Estimation

We followed the same method of generating scenarios as in Chapter 4. Thus, the waiting list data for scenarios was generated using the discrete-event simulation of End-Stage Liver Disease and organ allocation by Shechter et al. [155]. We modified the simulation model to capture snapshots of the national liver waiting list, and created scenarios using a sampling approach. Figure 4.1 depicts the sampling approach we used to create discrete scenarios using the simulation model of Shechter et al. [155]. See Section 4.4 for more details.

We used a constant value for an aggregate patient's probability of accepting a liver offer, i.e., $\tilde{q}_k$, across all scenarios. Using equation (5.3), and $q_k = 0.206$ and $J_k = 10$, from Section 4.4, we get $\tilde{q}_k = 0.9004$, for $k = 1, \ldots, K$. While creating aggregate patients for each scenario, we used a uniform distribution, and hence, set $\phi_m = \frac{1}{J_k}$ for $k = 1, \ldots, K, m \in M_k$.

5.4 COMPUTATIONAL RESULTS

During our initial computational experiments, we tested the performance of different geographic decomposition schemes on 10 batches of 200 scenarios. The initial set of regions was the set of all enumerated regions with no more than 3 OPOs in all cases. We used a UNIX machine with AMD Opteron 240 processor and 3.8 GB RAM for our computational tests, and coded the optimization algorithms using C++ and the CPLEX 9.0 Callable Library.

First, we tested the performance of geographic decomposition schemes that were introduced in Chapter 3, i.e., 20_10, 20_11, 20_12 and 20_15. Table 5.2 shows the summary of our initial analysis. Figure 5.2 shows the percentage increase in the objective value and CPU time for each geographic decomposition scheme when compared to 20_10. Recall that a geographic decomposition scheme ID, K_L, shows that the scheme includes K pricing problems each containing L OPOs. We already discussed how the size of the pricing problems grow with the number of OPOs included in the associated region cover. Table 5.2 and Figure 5.2 show that the CPU time is very sensitive to the number of OPOs in each region cover, i.e., L. After observing the relatively small increase in objective values at the expense of big

Table 5.2: Summary of initial tests with $B = 10, K = 200$.

Geographic Decomposition Scheme ID	Average objective	Average CPU time (sec.)
20_10	2,297.48	290.21
20_11	2,299.52	781.06
20_12	2,303.56	1,953.25
20_15	2,306.55	19,137.28
30_10	2,306.19	392.73

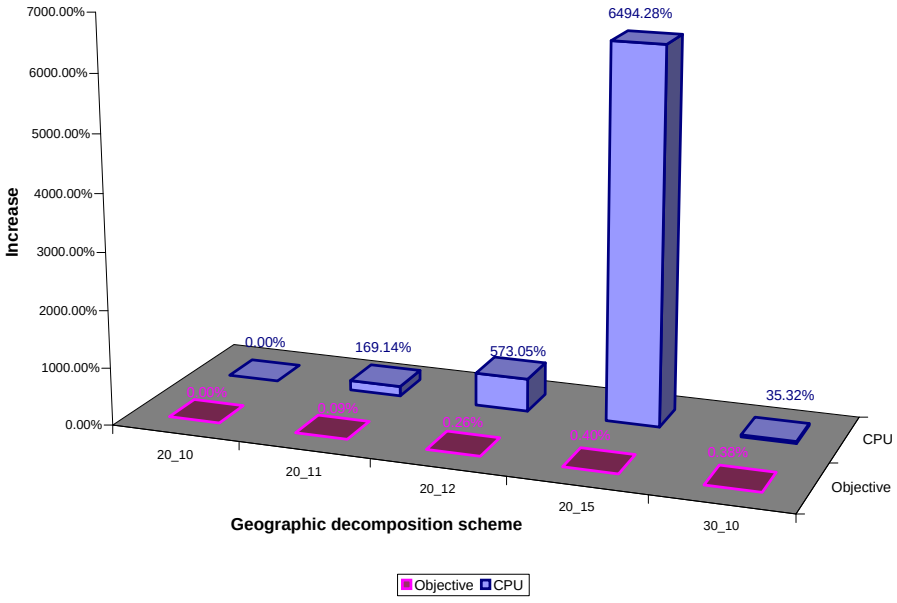


Figure 5.2: A comparison of percentage changes in objective values and run times for different geographic schemes when compared to 20_10.

jumps in run times when using geographic decomposition schemes with increasing L, and discovering that the maximum number of OPOs selected for a region in the optimal configurations obtained by our initial tests was 8, we decided to design and test a new scheme, i.e., 30_10. Table 5.2 and Figure 5.2 also show the performance of 30_10 under the same batches of scenarios. As can be seen from Figure 5.2, 30_10 brought almost the same amount of gain in the optimal objective value as 20_15 with a much lower increase in CPU time, and also generated better solutions faster than schemes 20_11 and 20_12. Hence, we chose 30_10 for our second set of tests with larger batches of scenarios.

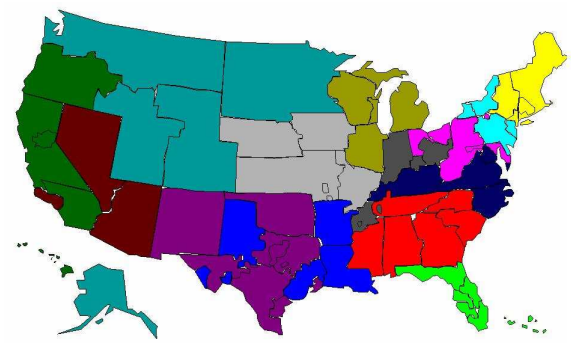
For the second set of computational tests we generated $B = 20$ batches of $K = 1,000$ scenarios according to the process explained in Section 4.4. Once again, we used the set of all contiguous OPOs with up to 3 OPOs as the initial set of columns for the master problem. The average CPU time for a batch of 1,000 scenarios using geographic decomposition scheme 30_10 was 1 hour 13 minutes and 47 seconds. Recall from Section 4.5 that the average time to solve an SAA problem with 200 scenarios using the patient-based model of Chapter 4 with geographic decomposition scheme 20_10 was around 4 hours. Hence, as intended, the aggregate modeling framework provides greater flexibility in terms of computational testing with different geographic decomposition schemes and larger batches of scenarios.

As we did in Section 4.5.1, we conducted statistical tests for obtaining estimates on solution qualities. We refer to Section 4.5.1 for the notation and details on the optimality gap estimates. Details on the optimal solutions found by solving each SAA problem in the second set of tests along with estimated optimality gaps can be found in Table 5.3. Table 5.3 shows that considerably tight optimality gaps of around 0.5% were obtained using the SAA method with $B = 20, K = 1,000$ under geographic decomposition scheme 30_10. Furthermore, Figures 5.3 through 5.6 show the regional configurations that correspond to these candidate solutions. Note that the maximum number of OPOs in any region shown in these figures is 8.

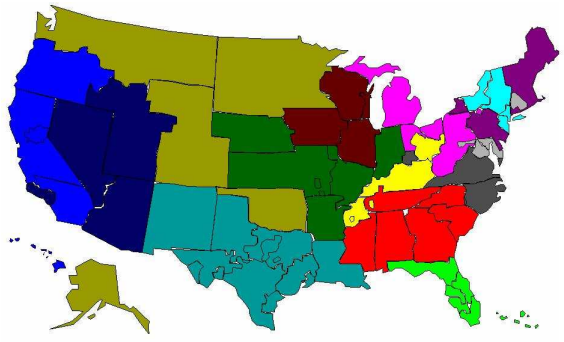
As an attempt to investigate the validity of the solutions obtained, we simulated the regional configurations shown in Figures 5.3 through 5.6 using the simulation model of Shechter et al. [155]. We simulated the liver allocation system from 1996 to the end of 2002, and ran 20 replications of each configuration using a warm-up period of 3 years. In order to compare

Table 5.3: Estimating the optimality gap for candidate solutions using geographic decomposition scheme 30_10 with $B = 20, K = 1,000$.

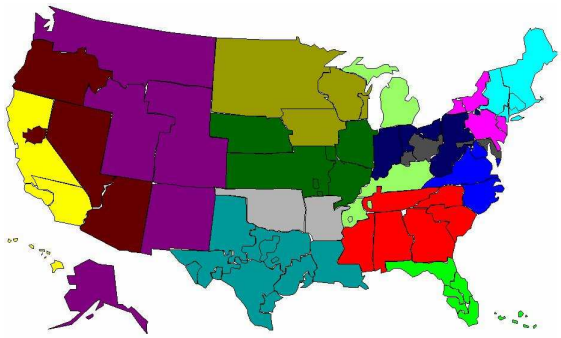
Batch (b)	Map ref.	$\bar{g}_K^B(\hat{x}_K^b)$	$\hat{v}_K^b$	Opt. gap ($\bar{v}_K^B - \bar{g}_K^B(\hat{x}_K^b)$)		95% CI				
						Absolute		% of $\bar{v}_K^B$		
				Mean	Var.	LL	UL	LL	Mean	UL
1	5A	2,253.17	2,271.17	15.06	1.66	10.03	20.09	0.44%	0.66%	0.89%
2	5B	2,252.39	2,253.09	15.84	1.64	10.83	20.84	0.48%	0.70%	0.92%
3	5C	2,249.30	2,265.93	18.93	1.54	14.09	23.77	0.62%	0.83%	1.05%
4	5D	2,255.26	2,248.37	12.96	1.28	8.55	17.38	0.38%	0.57%	0.77%
5	5E	2,258.37	2,270.38	9.86	1.29	5.42	14.30	0.24%	0.43%	0.63%
6	5F	2,254.16	2,298.94	14.06	1.54	9.21	18.92	0.41%	0.62%	0.83%
7	5G	2,254.82	2,272.49	13.40	1.40	8.78	18.02	0.39%	0.59%	0.79%
8	5H	2,253.51	2,282.39	14.72	1.25	10.36	19.08	0.46%	0.65%	0.84%
9	5I	2,253.07	2,268.02	15.16	1.13	11.01	19.31	0.49%	0.67%	0.85%
10	5J	2,254.91	2,246.37	13.32	1.22	9.01	17.63	0.40%	0.59%	0.78%
11	5K	2,255.02	2,255.80	13.21	1.50	8.42	17.99	0.37%	0.58%	0.79%
12	5L	2,252.69	2,265.90	15.53	1.72	10.42	20.65	0.46%	0.68%	0.91%
13	5M	2,253.94	2,234.03	14.29	2.00	8.76	19.81	0.39%	0.63%	0.87%
14	5N	2,259.43	2,298.54	8.80	1.00	4.89	12.71	0.22%	0.39%	0.56%
15	5O	2,256.47	2,278.26	11.76	1.12	7.63	15.89	0.34%	0.52%	0.70%
16	5P	2,255.82	2,282.34	12.41	1.36	7.85	16.96	0.35%	0.55%	0.75%
17	5Q	2,252.23	2,271.12	16.00	2.75	9.52	22.47	0.42%	0.71%	0.99%
18	5R	2,254.15	2,255.17	14.08	1.29	9.65	18.52	0.43%	0.62%	0.82%
19	5S	2,258.40	2,249.81	9.83	1.88	4.47	15.19	0.20%	0.43%	0.67%
20	5T	2,255.47	2,296.43	12.76	2.28	6.87	18.65	0.30%	0.56%	0.82%



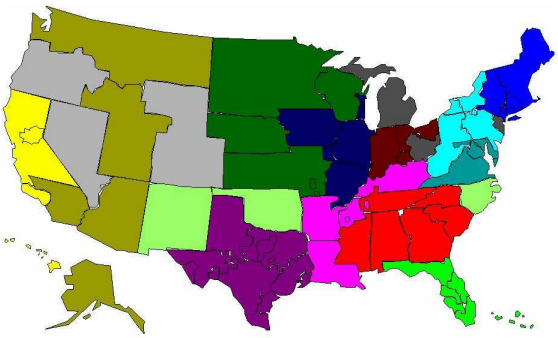
(5A)



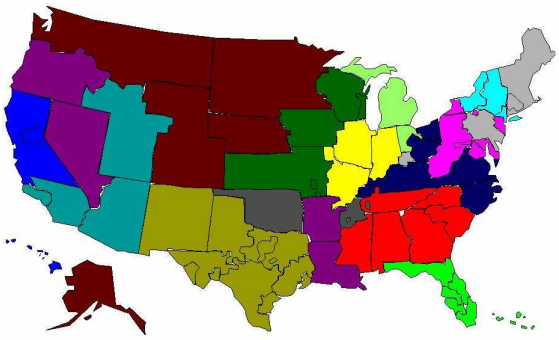
(5B)



(5C)



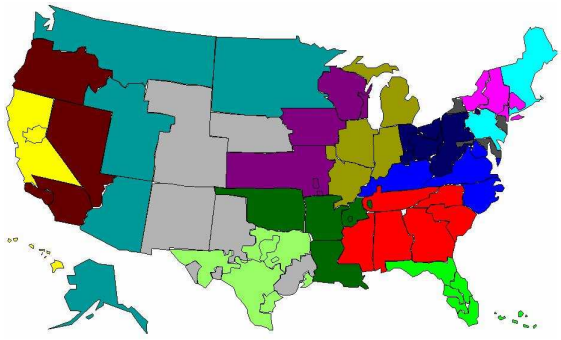
(5D)



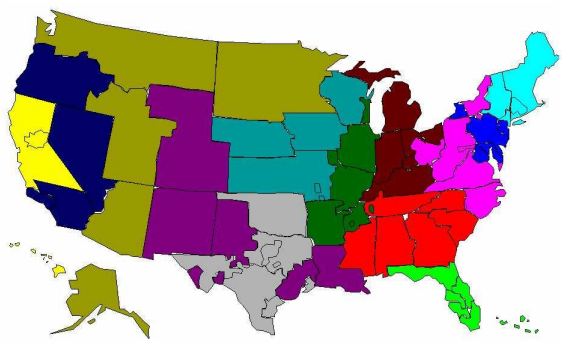
(5E)

Figure 5.3: Maps of regions obtained with geographic decomposition scheme 30_10 with $B = 20, K = 1,000$.

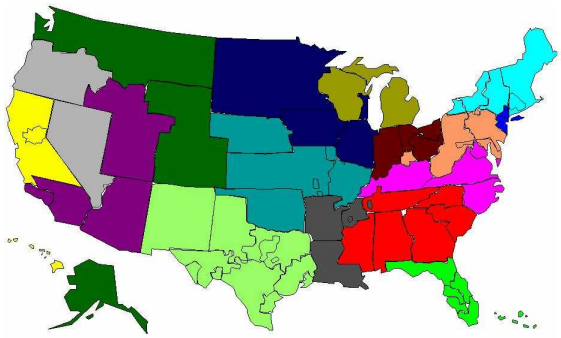
the performance of our alternative configurations with the current system, we conducted paired t tests. The results of the analysis can be seen in Table 5.4, and Figures 5.7 and 5.8. As in Chapter 4, the null hypothesis of these tests is that the average benefits gained per transplant are equal between an alternative configuration and the current regional configuration. Figure 5.7 displays the 95% confidence intervals around the mean difference



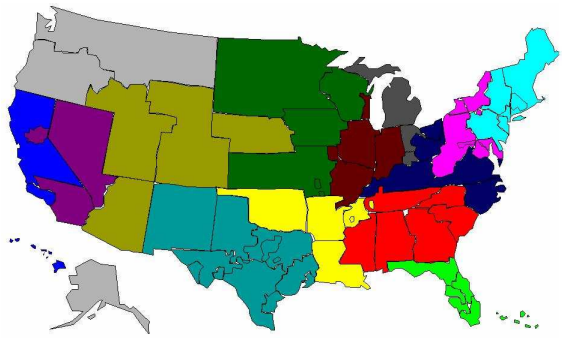
(5F)



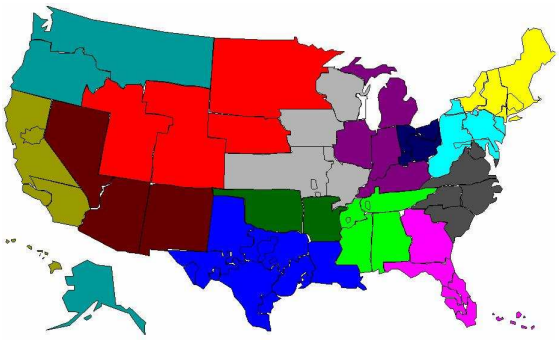
(5G)



(5H)



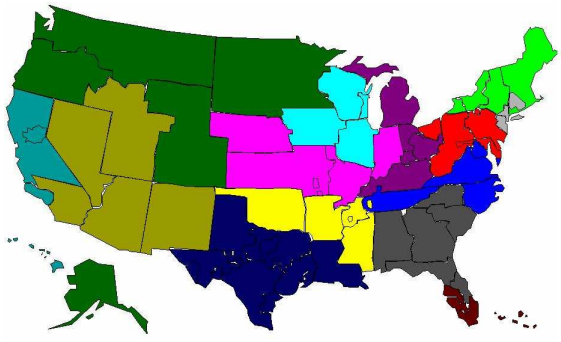
(5I)



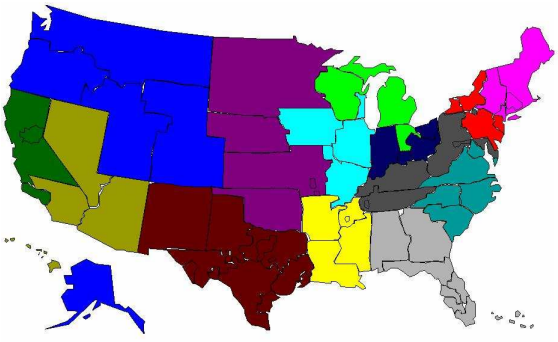
(5J)

Figure 5.4: Maps of regions obtained with geographic decomposition scheme 30_10 with $B = 20, K = 1,000$ continued.

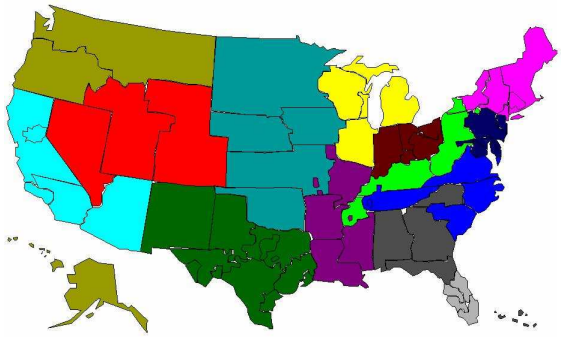
obtained in the expected benefit of each configuration when compared to the performance of the current system while Figure 5.8 shows the same results as a percentage of the current system. Since all $p-$values are 0, we conclude that the proposed regional configurations bring significant increases to the expected benefits gained from liver transplants.



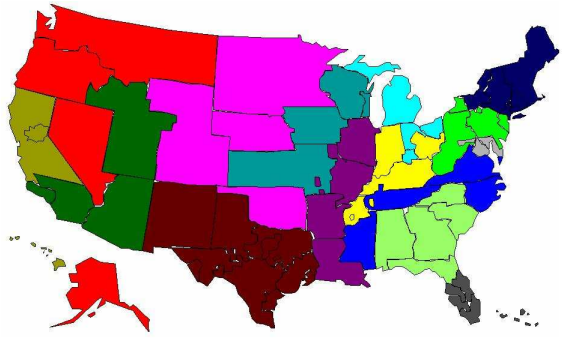
(5K)



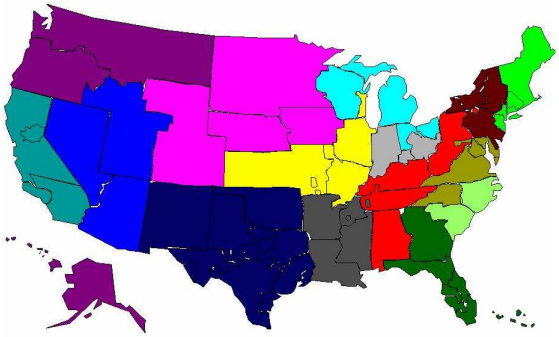
(5L)



(5M)



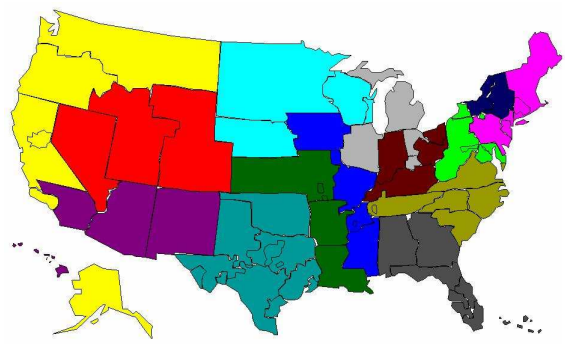
(5N)



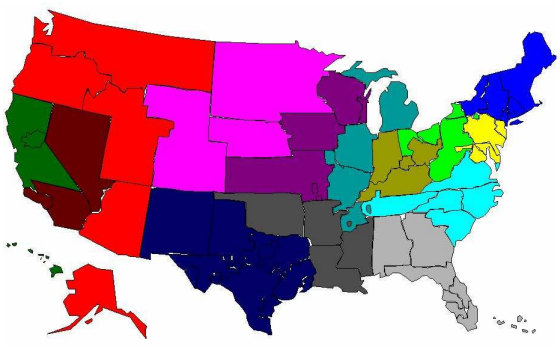
(5O)

Figure 5.5: Maps of regions obtained with geographic decomposition scheme 30_10 with $B = 20, K = 1,000$ continued.

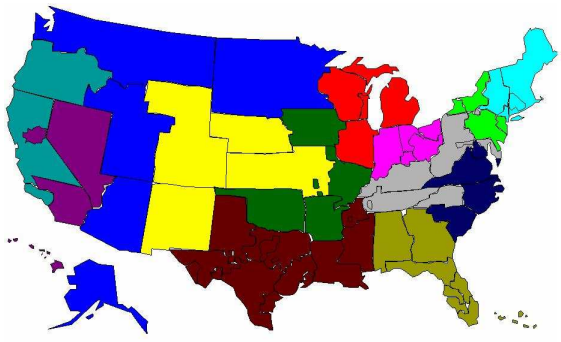
Comparing Tables 4.5 and 5.4 shows that the solutions obtained using the patient-based model of Chapter 4 and OPO-based model introduced in this chapter resulted in a similar performance in terms of the expected life-time gained by liver transplants. We already mentioned that the OPO-based model is much faster than its patient-based counterpart, solving a sample with $K = 1,000$ under geographic decomposition scheme 30_10 in a little



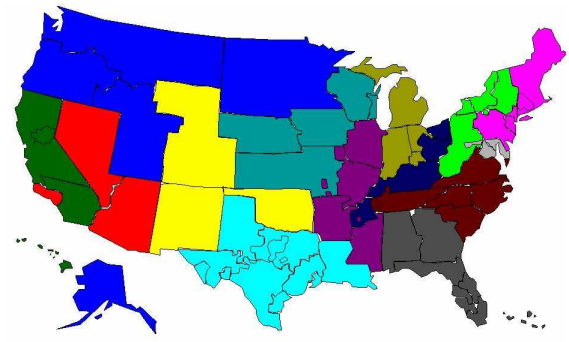
(5P)



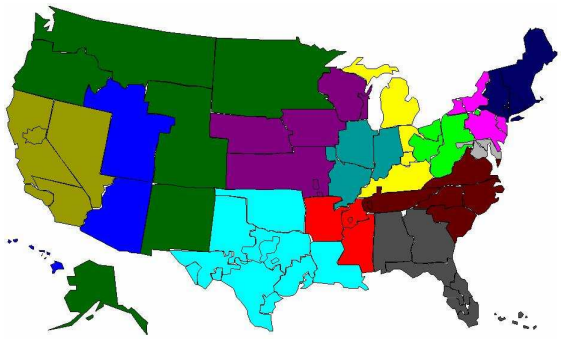
(5Q)



(5R)



(5S)



(5T)

Figure 5.6: Maps of regions obtained with geographic decomposition scheme 30_10 with $B = 20, K = 1,000$ continued.

over an hour on the average when compared to the 4 hour average solution time for the patient-based model under scheme 20_10 with $K = 200$. However, when we look at the candidate configurations listed in Tables 4.5 and 5.4, we observe that the configuration that performed best in simulation tests was regional configuration 4J of Figure 4.3, i.e., a solution found using the patient-based model. On the average, the solutions provided in Chapter 4

Table 5.4: Paired t tests and confidence intervals on the difference in expected life-days gained per transplant: Regional configurations of Figures 5.3 through 5.4 vs. current system

Regional Configuration	Paired Differences					t	$p-$value
	Mean	Standard Deviation	Standard Error	95% CI LL	95% CI UL		
5A	149.91	33.18	7.42	134.38	165.44	20.21	0.0000
5B	167.46	47.46	3.77	159.56	175.36	44.36	0.0000
5C	167.61	40.10	3.01	161.31	173.91	55.67	0.0000
5D	181.08	40.44	2.81	175.19	186.96	64.37	0.0000
5E	166.51	33.44	2.55	161.16	171.86	65.18	0.0000
5F	166.90	38.77	3.07	160.49	173.32	54.45	0.0000
5G	174.91	45.65	3.84	166.87	182.95	45.52	0.0000
5H	160.49	36.51	3.89	152.34	168.65	41.21	0.0000
5I	167.77	36.67	2.58	162.36	173.17	64.92	0.0000
5J	170.07	44.54	3.44	162.88	177.27	49.50	0.0000
5K	173.13	49.52	4.27	164.20	182.07	40.55	0.0000
5L	164.90	41.47	3.25	158.11	171.70	50.81	0.0000
5M	173.77	53.34	4.36	164.65	182.89	39.89	0.0000
5N	158.42	42.05	3.07	151.99	164.84	51.64	0.0000
5O	162.45	39.56	3.39	155.34	169.55	47.86	0.0000
5P	170.50	40.03	4.32	161.46	179.54	39.48	0.0000
5Q	170.41	39.69	3.54	163.01	177.82	48.17	0.0000
5R	177.59	44.65	3.39	170.49	184.69	52.36	0.0000
5S	175.18	43.99	3.20	168.49	181.88	54.76	0.0000
5T	146.24	41.20	3.15	139.64	152.85	46.37	0.0000

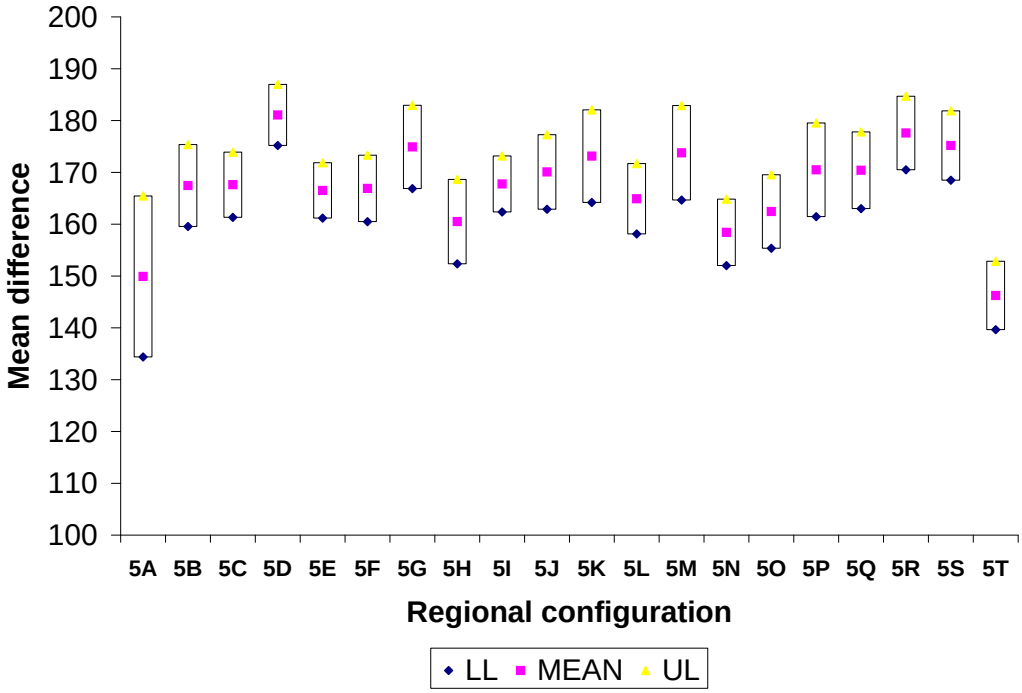


Figure 5.7: 95% confidence intervals around the mean difference in the expected life-days gained per transplant. Detailed results can be seen in Table 5.4.

and this section resulted in improvements of 4.14% and 4.93% in the expected life-time gained per transplant, respectively. The sample of scenarios used while generating configuration 4J of Figure 4.3 may possibly be a lucky choice of 200 scenarios.

Another interesting observation is that when Figures 4.2, 4.3 and 5.3 through 5.6 are examined, we observe that the patient-based model of Chapter 4 is more likely to produce noncontiguous configurations whereas, due to the smoothing of data through aggregation, the OPO-based approach produces contiguous solutions more frequently.

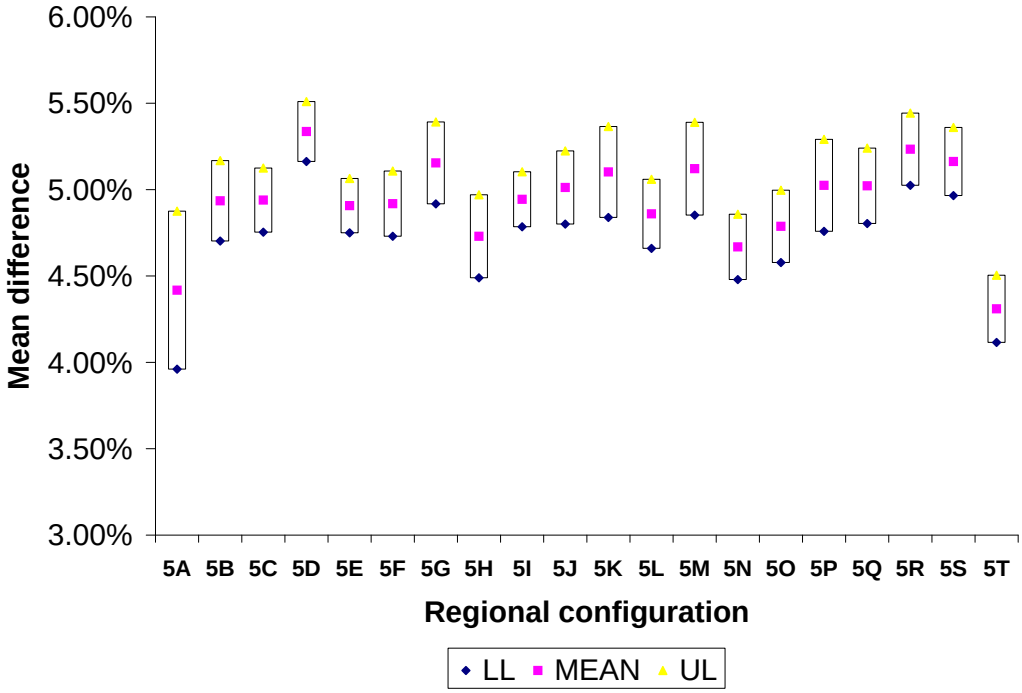


Figure 5.8: 95% confidence intervals around the mean difference in the expected life-days gained per transplant as percentage of the performance of the current system. Detailed results can be seen in Table 5.4.

5.5 CONCLUSIONS

In this chapter, we modified the patient-based stochastic mixed-integer programming model of Chapter 4 in order to develop a more computationally tractable framework for designing liver transplant regions under uncertainty. We preserved the same objective, i.e., maximizing the expected life-time gained by transplants at the regional level. We kept the main modeling structure and assumptions of Chapter 4 intact while aggregating the patient-related data in every scenario in an attempt to scale down the size of the problem.

We tested the performances on various geographic decomposition schemes on 10 samples of 200 scenarios, and observed that the solution time was driven by the size of region covers employed. We conducted a second set of tests using geographic decomposition scheme 30_10, which appeared to produce a good balance of solution quality and CPU time, on larger samples. Our tests resulted in tight optimality gap estimates, averaging around 0.5%.

We also simulated the regional configurations obtained using the simulation model of Shechter et al. [155]. The results show that our proposed regions are capable of improving the expected outcomes of liver transplant by an average of around 5%.

A possible direction for future research is testing different distributions of parameter ϕ_m, which shows the probability of patient m being offered the organ if the organ was only offered in her own OPO. In this chapter, we used a uniform distribution, and hence, created aggregate patients by averaging the data for the top patients in each OPO with an equal weight for each patient.

Incorporating an equity measure for the liver allocation system under a stochastic programming model is also left for future research. The models that patient- and OPO-based two-stage stochastic programming models that we introduced in this dissertation only focus on the efficiency of the system.

6.0 COLUMN GENERATION WITHIN THE L-SHAPED METHOD FOR STOCHASTIC LINEAR PROGRAMS

6.1 INTRODUCTION

In Chapters 4 and 5, we solved two-stage stochastic region-design problems using column generation. However, the fact that we modeled the ranking policies of UNOS within regions made second-stage decisions automatic and enabled a closed-form expression of the second-stage objectives. Hence, this eliminated the need for a mechanism combining column and cut generation to be used. If our models allowed for decisions on how to rank patients, then we would need to generate cuts and columns simultaneously for the master problem. Motivated by this observation, this chapter deals with the general question of how to design effective implementations of column generation methods within a two-stage stochastic programming framework.

We develop a method that incorporates column generation within the L-shaped method for solving two-stage stochastic linear programs [22, 174]. This method adds feasibility and optimality cuts generated from the second-stage subproblems based on the current columns in the master problem of the L-shaped method. It also generates columns for the master problem with respect to the first-stage constraints as well as existing feasibility and optimality cuts. We present different algorithmic strategies and prove finite convergence for a wide variety of approaches. We explore the performance of various algorithmic strategies that employ stabilization subroutines for strengthening both column and cut generation to effectively avoid degeneracy. To demonstrate the computational performances of the algorithmic approaches, we study two-stage stochastic versions of the well-known cutting stock and multi-commodity flow problems.

The primary contribution of this chapter is to apply combined column and cut generation to problems whose natural formulation involves a large number of columns. In this sense, the contribution may be seen as a modeling advance rather than a new solution technique that will perform well on problems for which column generation is inappropriate.

Two-stage stochastic linear programs are typically solved using decomposition methods that employ smaller subproblems that correspond to different scenarios. In the proposed approach, in addition to the second-stage scenario subproblems, we also have additional pricing subproblems to generate columns for the master problem. A comparison of the execution of the proposed approach and traditional stochastic programming is depicted in Figure 6.1.

The rest of the chapter is organized as follows: in Section 6.2, we discuss the theory behind our approach and perform the necessary reformulations for two-stage stochastic linear column generation models. In Section 6.3, we present the general framework for a class of algorithms incorporating column generation within the L-shaped method. We also discuss convergence issues and develop several algorithmic strategies involving different cut-aggregation, pricing and stabilization approaches. Then, in Section 6.4, we introduce two-stage stochastic versions of the cutting stock and multi-commodity flow problems, and analyze our computational experiments exploring the performance of our algorithmic approaches on these problems. Finally, we summarize our conclusions in Section 6.5.

6.2 THEORY AND REFORMULATIONS FOR COLUMN GENERATION WITHIN TWO-STAGE STOCHASTIC LINEAR PROGRAMS

Let n_i be the number of columns in stage i for $i = 1, 2$. Let $\{x | A'x \geq b', A''x \geq b'', x \geq 0\}$ be the set of feasible solutions for the first stage, where A' and A'' are known real-valued matrices of sizes $m_1' \times n_1$ and $m_1'' \times n_1$, respectively. Let b' and b'' be known vectors in $\mathbb{R}^{m_1'}$ and $\mathbb{R}^{m_1''}$, respectively. We divide the constraints that define this set into two such that one group of constraints ($A'x \geq b'$) is "hard" and the other ($A''x \geq b'', x \geq 0$) is "easy".

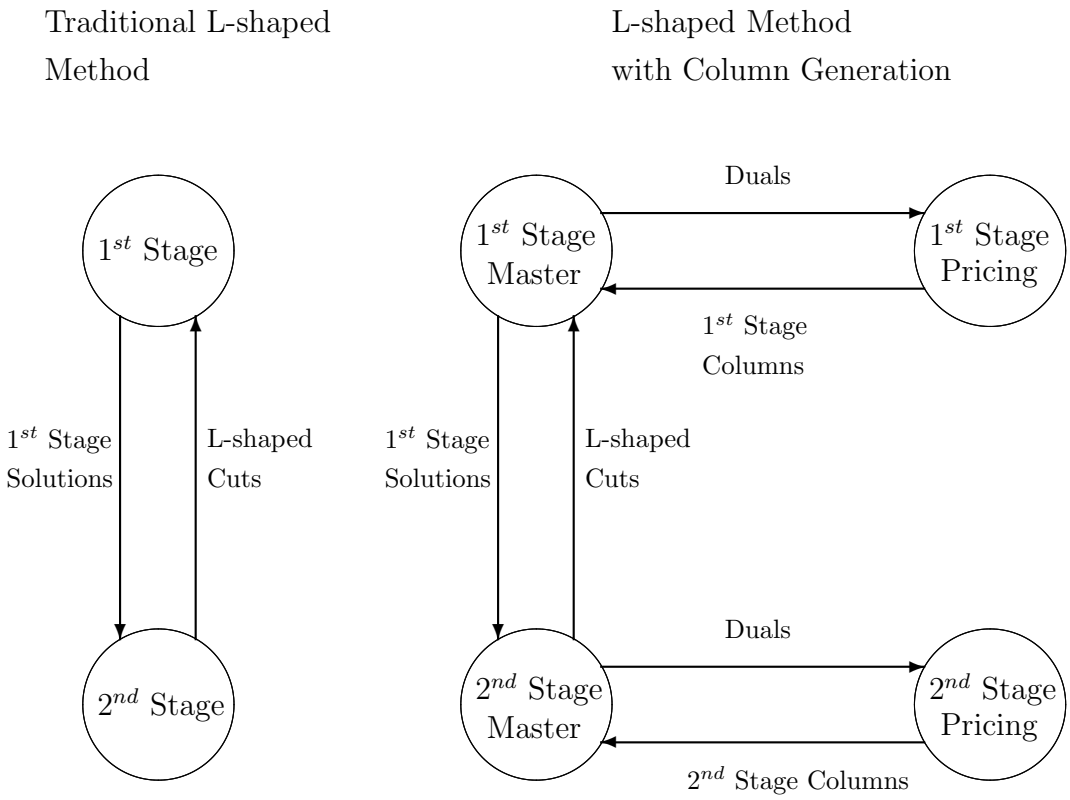


Figure 6.1: Schematic illustration of column generation within two-stage stochastic programming.

Let $\tilde{\xi}$ be a discrete random variable describing the uncertain parameters, and let Ξ be the finite support of $\tilde{\xi}$. For $k = 1, \ldots, K = |\Xi|$, let ξ^k describe the k^{th} element in Ξ, called a *scenario*, and let p^k be the probability that scenario ξ^k is realized. For each scenario $\xi^k \in \Xi$, let $\{y(\xi^k)|T(\xi^k)x + W'(\xi^k)y(\xi^k) \geq h'(\xi^k), W''(\xi^k)y(\xi) \geq h''(\xi^k), y(\xi^k) \geq 0\}$ be the set of feasible solutions for the second stage, where the *technology matrix*, $T(\xi^k)$, and the *recourse matrices*, $W'(\xi^k)$ and $W''(\xi^k)$, are of sizes $m_2' \times n_1$, $m_2' \times n_2$ and $m_2'' \times n_2$, respectively. Suppose the constraints that define this set can be divided into two, such that one group of constraints, $\{T(\xi^k)x + W'(\xi^k)y(\xi^k) \geq h'(\xi^k)\}$, is "hard" and the other, $\{W''(\xi^k)y(\xi) \geq h''(\xi^k), y(\xi^k) \geq 0\}$, is "easy".

A standard formulation of the extensive form of such a two-stage stochastic linear program, (P), following the formulations by Beale [15] and Dantzig [41] is as follows:

$$\min \; c^T x + \mathbb{E}_{\tilde{\xi}}[d(\tilde{\xi})^T y(\tilde{\xi})] \tag{6.1a}$$

$$\text{subject to} \quad A'x \geq b', \tag{6.1b}$$

$$A''x \geq b'', \tag{6.1c}$$

$$T(\xi^k)x + W'(\xi^k)y(\xi^k) \geq h'(\xi^k), \qquad \text{for } k = 1, \ldots, K, \tag{6.1d}$$

$$W''(\xi^k)y(\xi) \geq h''(\xi^k), \qquad \text{for } k = 1, \ldots, K, \tag{6.1e}$$

$$x \geq 0, \tag{6.1f}$$

$$y(\xi^k) \geq 0, \qquad \text{for } k = 1, \ldots, K, \tag{6.1g}$$

where c is a known vector in $\mathbb{R}^{n_1}$, and the "hard" first-stage constraints are defined by the constraint set (6.1b) whereas the "easy" first-stage constraints are constraints (6.1c) and (6.1f). Also, for each scenario, $d(\xi^k) \in \mathbb{R}^{n_2}$, $h(\xi^k) \in \mathbb{R}^{m_2}$, and the "hard" second-stage constraints are defined by the constraint set (6.1d) whereas the "easy" second-stage constraints are constraints (6.1e) and (6.1g). Piecing together the stochastic components of the problem, we obtain the scenario vector $\xi^k = (d(\xi^k)^T, h'(\xi^k)^T, h''(\xi^k)^T, T(\xi^k), W'(\xi^k), W''(\xi^k))$. For a given value of the first-stage vector x, the second-stage subproblem decomposes into K independent subproblems, one for each scenario.

The extensive form (6.1) is equivalent to the so-called *deterministic equivalent program*:

$$\begin{aligned} \min\ c^T x \quad &+ \quad \mathcal{Q}(x) \\ \text{subject to} \quad A'x \quad &\geq \quad b', \\ A''x \quad &\geq \quad b'', \\ x \quad &\geq \quad 0, \end{aligned} \tag{6.2}$$

where $\mathcal{Q}(x)$, the *expected recourse function*, is given by

$$\mathcal{Q}(x) \ = \ \mathbb{E}_{\tilde{\xi}} Q(x, \xi^k)), \tag{6.3}$$

and for every scenario ξ^k,

$$Q(x, \xi^k) \ = \ \min \ d(\xi^k)^T y \tag{6.4a}$$

$$\text{subject to} \qquad W'(\xi^k) y \ \geq \ h'(\xi^k) - T(\xi^k) x, \tag{6.4b}$$

$$W''(\xi^k) y \ \geq \ h''(\xi^k), \tag{6.4c}$$

$$y \ \geq \ 0. \tag{6.4d}$$

Algorithms for solving two-stage stochastic LPs exploit the fact that the expected recourse function $\mathcal{Q}(x)$ is convex [23]. Note that, for each $\xi^k \in \Xi$, we can express the polyhedron defined by the second-stage easy constraints, i.e., (6.4c) and (6.4d), as follows:

$$\Lambda_k = \{y \in \mathbb{R}^{n_2}_+ | W''(\xi^k) y \geq h''(\xi^k)\}. \tag{6.5}$$

Let $y_k^1, \ldots, y_k^{q_k}$ and $y_k^{q_k+1}, \ldots, y_k^{r_k}$ be the extreme points and extreme rays of Λ_k, respectively, and define

1. $\zeta_i(\xi^k) = d(\xi^k)^T y_k^i, \ i = 1, \ldots, r_k$, where $\zeta_i(\xi^k) \in \mathbb{R}$,
2. $\varrho_i(\xi^k) = W'(\xi^k) y_k^i, \ i = 1, \ldots, r_k$, where $\varrho_i(\xi^k) \in \mathbb{R}^{m_2'}$.

Then, by applying Minkowski's finite basis theorem [121] to the second-stage easy polyhedra, we can rewrite each second-stage subproblem as shown below:

$$Q(x,\xi^k) = \min \sum_{i=1}^{r_k} \zeta_i(\xi^k) u_i \tag{6.6a}$$

$$\text{subject to} \quad \sum_{i=1}^{r_k} \varrho_i(\xi^k) u_i \geq h'(\xi^k) - T(\xi^k)x, \tag{6.6b}$$

$$\sum_{i=1}^{q_k} u_i = 1, \tag{6.6c}$$

$$u_i \geq 0, \qquad \text{for } i = 1,\ldots,r_k. \tag{6.6d}$$

Now, for each scenario $\xi^k \in \Xi$, define the dual polyhedron

$$\begin{aligned}\Delta_k = \{\delta \in \mathbb{R}_+^{m_2'}, \delta^0 \in \mathbb{R}^1 | \delta^T \varrho_i(\xi^k) + \delta^0 \leq \zeta_i(\xi^k), \qquad & \text{for } i = 1,\ldots,q_k, \\ \delta^T \varrho_i(\xi^k) \leq \zeta_i(\xi^k), \qquad & \text{for } i = q_k+1,\ldots,r_k\}.\end{aligned} \tag{6.7}$$

Let $(\delta_1^k, \delta^{0k}_1),\ldots,(\delta_{l^k}^k, \delta^{0k}_{l^k})$ denote the extreme points of Δ_k, and let $(\delta^k_{l^k+1}, \delta^{0k}_{l^k+1}),\ldots,(\delta^k_{\mu^k}, \delta^{0k}_{\mu^k})$ denote the extreme rays of Δ_k. Then, applying the L-shaped reformulation [174] yields the following equivalent formulation:

$$\min \; c^T x \; + \sum_{k=1}^{K} \theta_k \tag{6.8a}$$

$$\text{subject to} \quad A'x \geq b', \tag{6.8b}$$

$$A''x \geq b'', \tag{6.8c}$$

$$(\delta_j^k)^T (h'(\xi^k) - T(\xi^k)x) + \delta^{0k}_j) \leq 0, \qquad \text{for } k = 1,\ldots,K,\ j = l^k+1,\ldots,\mu^k, \tag{6.8d}$$

$$p^k \left((\delta_j^k)^T (h'(\xi^k) - T(\xi^k)x) + \delta^{0k}_j) \right) \leq \theta_k, \qquad \text{for } k = 1,\ldots,K,\ j = 1,\ldots,l^k, \tag{6.8e}$$

$$x \geq 0, \tag{6.8f}$$

where constraint sets (6.8e) and (6.8d) represent the optimality and feasibility cuts generated by the L-shaped subproblems, respectively.

In the formulation above, constraint sets (6.8c) and (6.8f) are easy constraints, and sets (6.8b), (6.8e) and (6.8d) are hard constraints. Consider the polyhedron defined by constraint sets (6.8c) and (6.8f), i.e. the easy constraints:

$$\Lambda = \{x \in \mathbb{R}^{n_1}_+ | A''x \geq b''\}, \tag{6.9}$$

and let $x^1, \dots x^q$ and $x^{q+1}, \dots, x^r$ be the extreme points and extreme rays of Λ. By using Minkowski's finite basis theorem [121] we can rewrite Λ as

$$\Lambda = \left\{ x = \sum_{i=1}^{r} z_i x^i \,\middle|\, z \in \mathbb{R}^r_+, \sum_{i=1}^{q} z_i = 1 \right\}. \tag{6.10}$$

By rewriting x in the problem defined in (6.8) in terms of z_i and x^i, we get

$$\min \; c^T \left(\sum_{i=1}^{r} z_i x^i \right) + \sum_{k=1}^{K} \theta_k \tag{6.11a}$$

$$\text{subject to} \quad A' \left(\sum_{i=1}^{r} z_i x^i \right) \geq b', \tag{6.11b}$$

$$(\delta^k_j)^T \left[h'(\xi^k) - T(\xi^k) \left(\sum_{i=1}^{r} z_i x^i \right) \right] + \delta^{0k}_j \leq 0, \quad \text{for } k = 1, \dots, K, j = l^k + 1, \dots, \mu^k, \tag{6.11c}$$

$$p^k \left[(\delta^k_j)^T \left(h'(\xi^k) - T(\xi^k) \left(\sum_{i=1}^{r} z_i x^i \right) \right) + \delta^{0k}_j \right] \leq \theta_k, \quad \text{for } k = 1, \dots, K, j = 1, \dots, l^k, \tag{6.11d}$$

$$\sum_{i=1}^{q} z_i = 1, \tag{6.11e}$$

$$z_i \geq 0, \quad \text{for } i = 1, \dots, r. \tag{6.11f}$$

Define

1. $f_i = c^T x^i, i = 1, \dots, r$, where $f_i \in \mathbb{R}$,
2. $g_i = A' x^i, i = 1, \dots, r$, where $g_i \in \mathbb{R}^{m'_1}$,
3. $\gamma^k_i = T(\xi^k) x^i, \; i = 1, \dots, r, \; k = 1, \dots, K$, where $\gamma^k_i \in \mathbb{R}^{m_2}$.

Then the problem above can be rewritten as (P'):

$$\min \sum_{i=1}^{r} f_i z_i + \sum_{k=1}^{K} \theta_k \tag{6.12a}$$

$$\text{subject to} \quad \sum_{i=1}^{r} g_i z_i \geq b', \tag{6.12b}$$

$$(\delta_j^k)^T \left(h(\xi^k) - \sum_{i=1}^{r} \gamma_i^k z_i \right) + \delta^{0k}_j \leq 0, \quad \text{for } k = 1, \ldots, K, j = l^k + 1, \ldots, \mu^k, \tag{6.12c}$$

$$p^k \left[(\delta_j^k)^T \left(h(\xi^k) - \sum_{i=1}^{r} \gamma_i^k z_i \right) + \delta^{0k}_j \right] \leq \theta_k, \quad \text{for } k = 1, \ldots, K, j = 1, \ldots, l^k, \tag{6.12d}$$

$$\sum_{i=1}^{q} z_i = 1, \tag{6.12e}$$

$$z_i \geq 0, \quad \text{for } i = 1, \ldots, r. \tag{6.12f}$$

Thus, we have just proven the following theorem:

Theorem 6.1. *The reformulation (P') is an equivalent representation of the original problem (P).*

Formulation (P') is simply derived by applying the L-shaped and two Dantzig-Wolfe reformulations to (P).

6.3 ALGORITHMS FOR COMBINING DANTZIG-WOLFE DECOMPOSITION AND THE L-SHAPED METHOD

6.3.1 Main Algorithmic Approach

Note that (P') assumes that all extreme points and rays of the polyhedra Λ, Λ_k and Δ_k, for $k = 1, \ldots, K$ are known. However, computing all of these requires huge amounts of time and storage, and the resulting number of variables and constraints can be extremely large. Instead, an iterative procedure combining the column generation approach and the L-shaped method can be followed. This procedure is composed of a restricted master problem (RMP) which includes subsets of the extreme points and rays of the polyhedra Λ, Λ_k and Δ_k, for

$k = 1, \ldots, K$, at any point in the algorithm. Consequently, we will have subsets of the z variables and subsets of optimality and feasibility cuts in the RMP, out of the total possible number of z variables and constraints of the original problem (P'). New columns, i.e. new z variables, and new optimality and feasibility cuts will be generated as needed, through the use of subproblems. To clarify the notation, we describe our algorithm in the context of the single-cut L-shaped method; the extension to the multi-cut version [22] is straightforward. A simple framework of a possible algorithm to incorporate column generation within the L-shaped method can be given as follows:

1. Solve the RMP.
2. Search for an L-shaped cut or a favorable first-stage column (solve the corresponding second-stage subproblems to optimality when searching for a cut).
3. a. If none exists, the current solution is optimal.
   b. Otherwise, add the cut or column generated to the RMP and go to Step 1.

Figure 6.2 shows a flow chart for the general algorithm. The details of the algorithm are explained below:

- ***Initialization.***
  - Set $s_f = s_o = \nu = 0$. The first two are counters for feasibility and optimality cuts in the RMP, and the last one is the iteration counter.
  - Set $NeedCuts = NeedCols = True$. These two variables indicate whether the current RMP is optimal with respect to the first-stage columns and L-shaped cuts, respectively. The RMP is optimal with respect to the first-stage columns (L-shaped cuts) when $NeedCuts = False$ ($NeedCols = False$).
  - Let τ and φ denote the index sets of known extreme points and rays of the polyhedron Λ, respectively (i.e. the number of z variables equals $|\tau|+|\varphi|$). We assume $|\tau|+|\varphi| > 0$ at the beginning, and utilize a Phase-I type approach to generate initial columns in cases where constructing an initial feasible solution for the RMP is nontrivial.

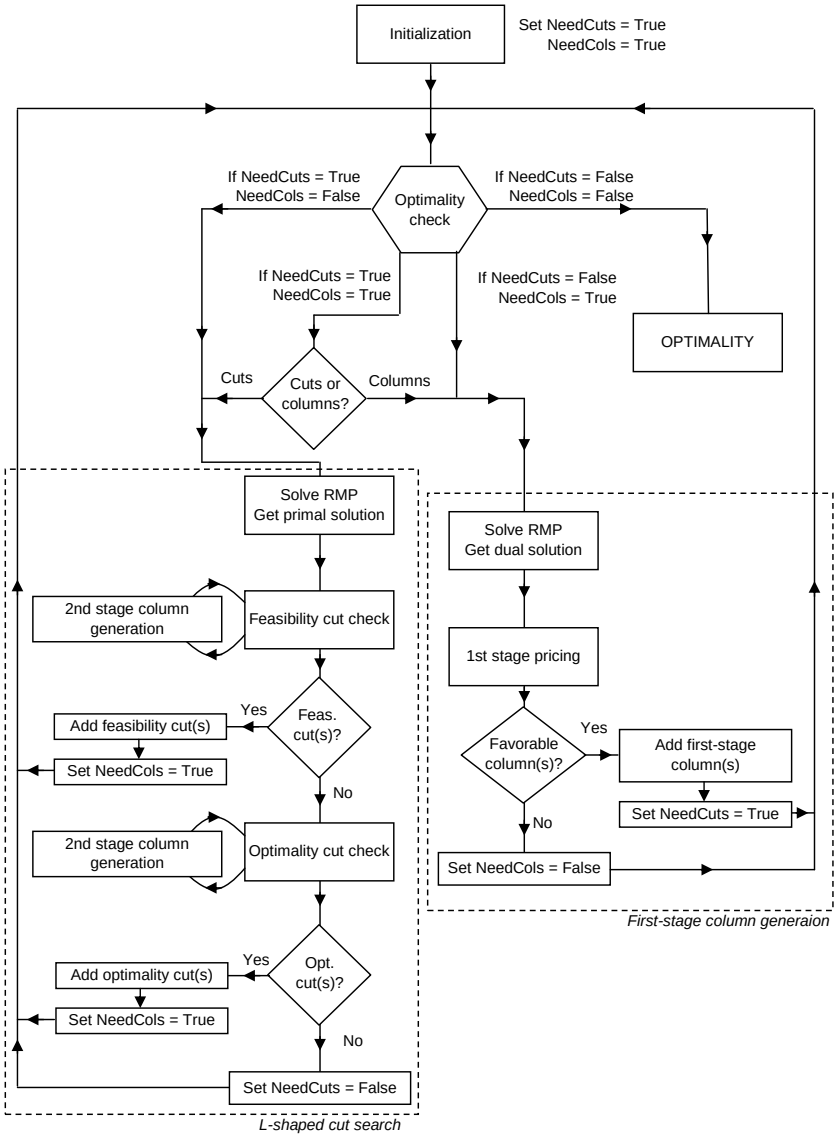


Figure 6.2: Flow chart for column generation within the L-shaped method.

– For every $k = 1, \ldots, K$, let $\bar{\tau}_k$ and $\bar{\varphi}_k$ denote the sets of known extreme points and rays of the easy second-stage polyhedron Λ_k, respectively. We assume $|\bar{\tau}_k| + |\bar{\varphi}_k| > 0$, for $k = 1, \ldots, K$, at the beginning, and utilize a Phase-I type approach to generate initial columns in cases where constructing initial feasible solutions for the second-stage subproblems is nontrivial.

- ***Optimality check.***

  This procedure checks whether the current solution is optimal. If not, it selects a procedure to use.

  – If $NeedCuts = NeedCols = True$, then go to either one of the procedures ***L-shaped cut generation*** or ***First-stage column generation*** depending on the choice of the user. The procedure L-shaped cut generation (First-stage column generation) may be interrupted by the Optimality check procedure after the addition of a cut (column), and depending on the user's preference on when to switch, the algorithm may resume the same procedure or switch to First-stage column generation (L-shaped cut generation).

  – If $NeedCuts = True$ and $NeedCols = False$, then go to the procedure ***L-shaped cut generation***.

  – If $NeedCuts = False$ and $NeedCols = True$, then go to the procedure ***First-stage column generation***.

  – If $NeedCuts = NeedCols = False$, then STOP; the current solution is OPTIMAL.

- ***L-shaped cut generation.***

  1. *Master problem.* Set $\nu = \nu + 1$. Solve the restricted master problem (RMP) below:

$$\begin{aligned}
\min f^T z \;+\; \theta & & & \text{(6.13a)}\\
\text{subject to} \quad g^T z &\geq b', & & \text{(6.13b)}\\
\bar{D}_\iota z &\geq \bar{d}_\iota, \qquad \text{for } \iota = 1, \ldots, s_f, & \text{(6.13c)}\\
\bar{E}_\iota z + \theta &\geq \bar{e}_\iota, \qquad \text{for } \iota = 1, \ldots, s_o, & \text{(6.13d)}\\
\sum_{i \in \tau} z_i &= 1, & \text{(6.13e)}\\
z_i &\geq 0, \qquad \text{for } i \in \tau \cup \varphi, & \text{(6.13f)}
\end{aligned}$$

where constraint sets (6.13c) and (6.13d) are the L-shaped feasibility and optimality cuts, respectively. Let (z^ν, θ^ν) denote the optimal solution to the RMP. If $s_o = 0$, then θ is assumed to be equal to $-\infty$ for iteration ν.

2. *Feasibility cut generation.* Set the scenario index $k = 1$.

   a. *Feasibility subproblem.* Solve the following linear program:

$$w' = \min \ \mathbf{1}^T v^+ - \mathbf{1}^T v^- \tag{6.14a}$$

$$\text{subject to} \sum_{i \in \bar{\tau}_k \cup \bar{\varphi}_k} \varrho_i(\xi^k) u_i + I^+ v^+ - I^- v^- \ \geq \ h'(\xi^k) - \Gamma^k z^\nu, \tag{6.14b}$$

$$\sum_{i \in \bar{\tau}_k} u_i \ = \ 1, \tag{6.14c}$$

$$u_i \ \geq \ 0, \quad \forall i \in \bar{\tau}_k \cup \bar{\varphi}_k, \tag{6.14d}$$

   where $\mathbf{1}$ is a vector of ones and Γ^k is the $m_2 \times (|\tau| + |\varphi|)$ matrix where the columns correspond to γ_i^k, $i \in \tau \cup \varphi$. Let δ_k and δ_k^0 denote the optimal dual variables corresponding to constraint sets (6.14b) and (6.14c), respectively.

   – If $w' = 0$ and $k = K$, go to step 3 (*optimality cut generation*) .

   – If $w' = 0$ and $k < K$, set $k = k + 1$ and go to step 2a (*feasibility subproblem*).

   – If $w' > 0$, no feasible solution can be generated by using the current elements of the sets $\bar{\tau}_k$ and $\bar{\varphi}_k$. However, this does not show that the whole subproblem is infeasible. We have to generate more extreme points and rays for the sets $\bar{\tau}_k$ and $\bar{\varphi}_k$. Go to step 2b (*second-stage column generation for the feasibility subproblem*).

   b. *Second-stage column generation for the feasibility subproblem.* Solve the following *second-stage pricing subproblem*:

$$\max \ \bar{c}_y = \delta_k^T W'(\xi^k) y \tag{6.15a}$$

$$\text{subject to} \quad W''(\xi^k) y \ \geq \ h''(\xi^k), \tag{6.15b}$$

$$y \ \geq \ 0. \tag{6.15c}$$

   – If $\bar{c}_y \leq -\delta_k^0$, then no column can improve the objective function, w', so the whole second-stage subproblem is infeasible. Go to step 2c (*feasibility cut insertion).*

– If $\bar{c}_y = \infty$, add a corresponding extreme ray of the polyhedron Λ_k to $\bar{\varphi}_k$, add the corresponding u variable to the feasibility subproblem, and return to step 2a (*feasibility subproblem*).

– If $-\delta_k^0 < \bar{c}_y < \infty$, add the corresponding extreme point of Λ_k to $\bar{\tau}_k$, add the corresponding u variable to the feasibility subproblem and return to step 2a (*feasibility subproblem*).

c. *Feasibility cut insertion.* Calculate the cut coefficients for existing variables in the RMP and the right-hand side:

$$\bar{D}_{s_f+1} = (\delta_k^\nu)^T \Gamma^k, \tag{6.16a}$$

$$\bar{d}_{s_f+1} = (\delta_k^\nu)^T h'(\xi^k) + \delta_k^0, \tag{6.16b}$$

and add a feasibility cut of the form (6.13c). Calculate and store the *implicit technology coefficients* that will be used to construct new columns:

$$\widetilde{D}_{s_f+1} = (\delta_k^\nu)^T T(\xi^k). \tag{6.17}$$

Set $s_f = s_f + 1$, $NeedCols = True$. **Go to procedure *Optimality check*.**

3. *Optimality cut generation.* Set the scenario index $k = 1$.

a. *Optimality subproblem.* Solve the following linear program:

$$Q(z^\nu, \xi^k) = \min \sum_{i \in \bar{\tau}_k \cup \bar{\varphi}_k} \zeta_i(\xi^k) u_i \tag{6.18a}$$

$$\text{subject to} \quad \sum_{i \in \bar{\tau}_k \cup \bar{\varphi}_k} \varrho_i(\xi^k) u_i \geq h'(\xi^k) - \gamma^k z^\nu, \tag{6.18b}$$

$$\sum_{i \in \bar{\tau}_k} u_i = 1, \tag{6.18c}$$

$$u_i \geq 0, \qquad \forall i \in \bar{\tau}_k \cup \bar{\varphi}_k. \tag{6.18d}$$

Let δ_k and δ_k^0 denote the optimal dual variables corresponding to constraint sets (6.18b) and (6.18c), respectively. If $Q(z^\nu, \xi^k)$ is unbounded, this shows that the original problem is unbounded and the algorithm terminates. Otherwise, go to step 3b (*second-stage column generation for the optimality subproblem*).

b. *Second-stage column generation for the optimality subproblem.* Solve the following *second-stage pricing subproblem*:

$$\min \bar{c}_y = (d(\xi^k)^T - \delta_k^T W'(\xi^k))y \tag{6.19a}$$

$$\text{subject to} \quad W''(\xi^k)y \geq h''(\xi^k), \tag{6.19b}$$

$$y \geq 0. \tag{6.19c}$$

- If $\bar{c}_y \geq \delta_k^0$, then if $k = K$, go to step 3c (*optimality cut insertion*), otherwise set $k = k + 1$ and go to step 3a (*optimality subproblem*) .
- If $\bar{c}_y = -\infty$, add a corresponding extreme ray of the polyhedron Λ_k to $\bar{\varphi}_k$, add the corresponding u variable to $Q(z, \xi^k)$, and return to step 3a (*optimality subproblem*).
- If $-\infty < \bar{c}_y < \delta_k^0$, add the corresponding extreme point of Λ_k to $\bar{\tau}_k$, add the corresponding u variable to the corresponding optimality subproblem and return to step 3a (*optimality subproblem*).

c. *Optimality cut insertion.* Calculate the cut coefficients for existing variables in the RMP and the right-hand side:

$$\bar{E}_{s_o+1} = \sum_{k=1}^{K} p^k (\delta_k)^T \Gamma^k, \tag{6.20a}$$

$$\bar{e}_{s_o+1} = \sum_{k=1}^{K} p^k ((\delta_k)^T h'(\xi^k) + \delta_k^0). \tag{6.20b}$$

Calculate and store the implicit technology coefficients that will be used to construct new columns:

$$\tilde{E}_{s_o+1} = \sum_{k=1}^{K} p^k (\delta_k)^T T(\xi^k). \tag{6.21}$$

- If $w^\nu = \bar{e}_{s_o+1} - \bar{E}_{s_o+1} z^\nu > \theta^\nu$, add the corresponding optimality cut of the form (6.13d) and set $s_o = s_o + 1$, $NeedCols = True$. **Go to procedure *Optimality check***.

– Otherwise, the RMP is optimal for the current set of z variables. So, no optimality cut can be added. Set $NeedCuts = False$. **Go to procedure *Optimality check***.

- ***First-stage column generation.***
  1. *Master problem.* Solve the RMP (6.13), get the optimal dual values and set $\nu = \nu+1$. Let $\hat{\pi}^\nu$, $\hat{\sigma}^\nu$, $\hat{\rho}^\nu$ and $\hat{\eta}^\nu$ represent the optimal dual variables corresponding to constraint sets (6.13b), (6.13c), (6.13d) and (6.13e), respectively.
  2. *First-stage pricing subproblem.* Solve the following linear program:

$$\min \bar{c}^\nu = \left(c^T - (\hat{\pi}^\nu)^T A' - \left[\sum_{\iota=1}^{s_f} \hat{\sigma}_\iota^\nu \tilde{D}_\iota \right] - \left[\sum_{\iota=1}^{s_o} \hat{\rho}_\iota^\nu \tilde{E}_\iota \right] \right) x \tag{6.22a}$$

$$\text{subject to} \quad A'' x \geq b'', \tag{6.22b}$$

$$x \geq 0. \tag{6.22c}$$

     – If $\bar{c}^\nu \geq \hat{\eta}^\nu$, the current set of columns in the RMP is optimal. Set $NeedCols = False$. **Go to procedure *Optimality check***.
     – Otherwise, if $\bar{c}^\nu < \hat{\eta}^\nu$ and is bounded, add the z column corresponding to this extreme point of Λ, i.e. x, to the RMP, update τ accordingly, set $NeedCuts = True$ and **go to procedure *Optimality check***. If $\bar{c}^\nu < \hat{\eta}^\nu$ is unbounded, find the corresponding extreme ray, add the necessary z variable, update φ accordingly, set $NeedCuts = True$ and **go to procedure *Optimality check***.

Note that the general algorithm explained above gives rise to a wide variety of algorithmic strategies by providing flexibility in the choice of when to search for L-shaped cuts or first-stage columns in the *Optimality Check* procedure.

Although the reformulation (P') assumes that all extreme points and rays of Δ_k, $\forall k = 1, \ldots, K$, are known, in the procedure explained above, for a given master solution z, we compute optimal dual vectors of the optimality subproblems for every scenario $\xi^k \in \Xi$ at a particular iteration and generate an optimality cut by using these dual extreme points. Similarly, we derive feasibility cuts based on the rays generated from an incomplete dual polyhedron. A natural question that arises is whether these cuts will be valid for the whole formulation or not. The following proposition answers in the affirmative:

Proposition 6.1. *The cuts derived by using dual information from the second-stage subproblems at a particular iteration of the algorithm are valid for the whole formulation.*

Proof. **Validity of feasibility cuts:** Assume we generate a feasibility cut at iteration ν from the feasibility subproblem for scenario ξ^k by using the dual vector $(\delta_k,\, \delta_k^0)$. Then, $(\delta_k,\, \delta_k^0)$ is a ray of the incomplete dual polyhedron for scenario ξ^k at iteration ν, namely $\Delta_{k\nu}$. So, $(\delta_k,\, \delta_k^0) \in \Delta_{k\nu}^0$, which is the recession cone of $\Delta_{k\nu}$. Since we have solved the corresponding feasibility subproblem to optimality, we know that no excluded column prices out favorably. So, the necessary reduced cost calculations for any excluded variable, $u_{\hat{i}}$, show that

$$\delta_k{}^T \varrho_{\hat{i}}(\xi^k) + \delta_k^0 \leq 0, \quad \text{if } u_{\hat{i}} \text{ corresponds to an extreme point of } \Lambda_k,$$

and,

$$\delta_k{}^T \varrho_{\hat{i}}(\xi^k) \leq 0, \quad \text{if } u_{\hat{i}} \text{ corresponds to an extreme ray of } \Lambda_k,$$

which indicate that all the remaining constraints of Δ_k^0, i.e. the recession cone of the complete dual polyhedron, Δ_k, which are not present in $\Delta_{k\nu}^0$, will automatically be satisfied by $(\delta_k,\, \delta_k^0)$. So, $(\delta_k,\, \delta_k^0) \in \Delta_k^0$. This proves that $(\delta_k,\, \delta_k^0)$ is a ray for Δ_k.

Validity of optimality cuts: Assume that at a particular iteration ν, we get $(\delta_k,\, \delta_k^0)$, for $k = 1, \ldots, K$, and generate the corresponding optimality cut. Pick an arbitrary k and consider $(\delta_k,\, \delta_k^0)$. Since it is the optimal dual solution to the second-stage subproblem for scenario ξ^k, it has to be an extreme point of the incomplete dual formulation of the corresponding iteration, namely $\Delta_{k\nu}$. Since $(\delta_k,\, \delta_k^0)$ was generated by solving the corresponding second-stage subproblem to optimality at iteration ν, we know that all the excluded u variables at that point have unfavorable, i.e. non-negative, reduced costs. The necessary reduced costs calculations for an excluded second-stage variable, say $u_{\hat{i}}$, then, show that

$$\delta_k{}^T \varrho_{\hat{i}}(\xi^k) + \delta_k^0 \leq \zeta_{\hat{i}}(\xi^k), \quad \text{if } u_{\hat{i}} \text{ corresponds to an extreme point of } \Lambda_k,$$

and,

$$\delta_k{}^T \varrho_{\hat{i}}(\xi^k) \leq \zeta_{\hat{i}}(\xi^k), \quad \text{if } u_{\hat{i}} \text{ corresponds to an extreme ray of } \Lambda_k,$$

which indicate that all the remaining constraints of Δ_k, which are not present in $\Delta_{k\nu}$, will automatically be satisfied by $(\delta_k,\, \delta_k^0)$. So, $(\delta_k,\, \delta_k^0) \in \Delta_k$.

Now, assume that $(\delta_k,\ \delta_k^0)$ is not an extreme point of Δ_k. Then there exist two points $r_1, r_2 \in \Delta_1$ such that $(\delta_k,\ \delta_k^0)$ can be written as a convex combination of r_1 and r_2. However, since $(\delta_k,\ \delta_k^0)$ is an extreme point of $\Delta_{k\nu}$, it cannot be expressed as a convex combination of any two points in $\Delta_{k\nu}$. So, it should be the case that $r_1 \notin \Delta_{k\nu}$ or $r_2 \notin \Delta_{k\nu}$ which is a contradiction since $\Delta_k \subseteq \Delta_{k\nu}$. Thus, $(\delta_k,\ \delta_k^0)$ is an extreme point of Δ_k. □

The following theorem shows that convergence can be shown for a wide variety of possible algorithmic strategies. The proof follows from the fact that there are a finite number of columns and L-shaped cuts.

Theorem 6.2. *As long as an algorithm adds at least one first-stage column or an L-shaped cut from an infeasible or optimal second-stage solution at every iteration without removing any columns or cuts, such an algorithm will converge to the optimal solution.*

Proof. Assume that solving the RMP at the beginning of an iteration generates the first-stage solution $(\hat{z}, \hat{\theta})$. Now, there are three possibilities:

1. If $(\hat{z}, \hat{\theta})$ violates any of the L-shaped feasibility or optimality cuts, i.e. (6.12d) and (6.12c), it will be cut off by generating a cut generated from the second-stage subproblems.

2. Otherwise, if $(\hat{z}, \hat{\theta})$ does not violate any L-shaped cuts but there exists at least one excluded first-stage variable that has a strictly negative reduced cost, such a variable will be generated by solving the first-stage pricing subproblem.

3. Otherwise, if $(\hat{z}, \hat{\theta})$ does not violate any L-shaped cuts and all excluded first-stage variables have non-negative reduced costs, $(\hat{z}, \hat{\theta})$ is the optimal solution to the whole problem.

Note that conditions 1 and 2 indicate that $(\hat{z}, \hat{\theta})$ is non-optimal. If $(\hat{z}, \hat{\theta})$ satisfies either condition 1 or 2, solving the RMP after the addition of the corresponding L-shaped cut or first-stage column is guaranteed to change $(\hat{z}, \hat{\theta})$. So, any algorithm checking the conditions stated above and updating the incumbent first-stage solution accordingly will converge finitely (since there are finitely many first-stage columns and L-shaped cuts, and added cuts/columns are not removed). Checking the above conditions and updating the first-stage solution simply involves adding at least one L-shaped cut or one improving first-stage column when $(\hat{z}, \hat{\theta})$ is not optimal. Hence, any algorithm that guarantees this will converge finitely. □

6.3.2 Algorithmic Strategies

In this section we explore different strategies to design algorithms that follow the main framework.

6.3.2.1 L-shaped Cut Generation The main algorithmic framework shown in Section 6.3.1 adopts a complete aggregation of the dual information over all scenarios and follows the single-cut L-shaped method [174]. An alternative strategy that can be implemented regarding cut generation is the multi-cut L-shaped method [22], which places one cut per scenario, if the cut is violated, at every iteration. Both the single and multi-cut versions of the L-shaped method have their advantages and disadvantages, and their relative performances are problem dependent [22, 67]. Hybrid cut-generation strategies that involve the aggregation of dual information from a number of scenarios to generate aggregate cuts try to take advantage of the strengths of both of these methods by adding multiple cuts per iteration while aiming at keeping the size of the master problem relatively small. These types of approaches have proven to be useful [167].

Aggregation approaches divide the scenario set Ξ into mutually exclusive and exhaustive sets of scenarios, say $S_1, S_2, ..., S_C$, where $S_1 \cup S_2 \cup \ldots S_C = \Xi$, and $S_i \cap S_j = \emptyset$ for $i \neq j$. Each scenario set produces one optimality cut per iteration through aggregation of dual information coming from individual members of the set. Within the context of the algorithm

explained in Section 6.3.1, the cut coefficients and implicit technology coefficients for an optimality cut ι from scenario set S_i can be calculated as $\bar{E}_\iota = \sum_{k \in S_i} p^k (\delta_k)^T \Gamma^k$, $\bar{e}_\iota = \sum_{k \in S_i} p^k ((\delta_k)^T h'(\xi^k) + \delta_k^0)$, and $\tilde{E}_\iota = \sum_{k \in S_i} p^k (\delta_k^\nu)^T T(\xi^k)$. In this chapter, in addition to the single and multi-cut L-shaped methods, we also test the performance of static cut-aggregation methods.

6.3.2.2 Column Generation In the main algorithmic framework of Section 6.3.1, one favorable column is generated, if available, each time a pricing problem is solved, and the pricing problem is solved to optimality at every iteration. In fact, any column with a favorable reduced cost will suffice for the algorithm to work, and numerous strategies involving column generation can be employed.

One possible approach would be heuristically generating a favorable column, or terminating the solution of the pricing problem whenever a favorable column has been found without fully optimizing it at every iteration. As long as approaches like these are utilized in a way that guarantees that all columns are implicitly considered when no favorable column is returned, the algorithm will converge. An alternative would be generating multiple columns, if available, each time the pricing problem is solved. For our computational tests, we also test the performances of column generation strategies involving returning the first favorable column generated throughout the solution of the pricing problem without fully optimizing it, and inserting all enumerated favorable columns while solving the pricing problem to optimality.

6.3.2.3 Switching Criteria We explore different strategies involving *switching criteria*, which determine when to search for cuts and when to search for first-stage columns. We may define different switching criteria in order to come up with algorithmic variants. Note that it is impossible to create an exhaustive list of all possible switching strategies. In this section we introduce a few classes of intuitive switching criteria.

At an arbitrary iteration, the algorithm may either generate more columns for the RMP or generate L-shaped cuts from the second-stage subproblem. An algorithm could generate all favorable first-stage columns before it considers the second stage. It could then generate

all possible feasibility and optimality cuts in the second stage before returning to the first stage. This corresponds to an exhaustive switching criterion. However, because Benders' decomposition and Dantzig-Wolfe decomposition exhibit "tailing off" in practice [115], other approaches may be more effective. The efficiency of generating columns versus cuts is likely to depend on the difficulty of the first-stage pricing subproblem and second-stage subproblems.

Note that we still assume that each second-stage subproblem is solved to optimality for a given first-stage solution. That is, all necessary columns for a second-stage subproblem will be added until there are no favorable columns left when solving a second-stage subproblem at any iteration of the algorithm. So, the procedure of generating second-stage columns has nothing to do with the switching criterion. This ensures the validity of the L-shaped cuts derived from the second-stage subproblems, as discussed in Proposition 6.1. Therefore, through the remainder of this section, we will refer to generating first-stage columns when talking about column generation and assume that all favorable second-stage columns are added while solving a subproblem to add L-shaped cuts to the RMP.

In our computational experiments, we started the algorithm with the L-shaped cut search. A *major iteration* begins when the RMP is solved to get the primal optimal values before adding any cuts during that major iteration. Then, the search for cuts begins while the set of first-stage columns is held fixed. After that, the set of cuts becomes fixed and the search for favorable first-stage columns is performed. The major iteration ends when switching back from searching for first-stage columns to cuts occurs. So multiple cuts and first-stage columns are typically generated during a major iteration. A *minor iteration* means that the RMP is solved to get either the primal or dual optimal solutions in order to carry out one iteration of cut or column generation based on those solutions.

Some examples of switching criteria are as follows:

- **All-cuts-all-columns:** During a major iteration, add all possible cuts and all possible first-stage columns.
- **All-cuts-one-column:** During a major iteration, add all possible cuts but add at most one first-stage column.
- **One-cut-all-columns:** During a major iteration, add at most one cut but all favorable first-stage columns should be added.

- **Threshold approaches:** During a major iteration, the number of cuts and first-stage columns added depends on the objective function changes caused by the previously added cuts or first-stage columns at the same major iteration.
  - **Comparing to the first cut/column:** If the objective change caused by the last cut added is below a certain percentage $\mathcal{P}$ of the objective change resulting from the addition of the first cut during the same major iteration, stop searching for cuts and switch to columns. Switching back from first-stage columns to cuts, in a similar manner, depends on the objective value change caused by the addition of the first first-stage column added during that particular major iteration.
  - **Comparing to the best cut/column:** If the objective change caused by the last cut added is below a certain percentage $\mathcal{P}$ of the objective change resulting from adding the best cut during the same major iteration (i.e. the cut that caused the maximum increase in the objective at that major iteration), stop searching for cuts and switch to columns. Switching back from first-stage columns to cuts, similarly, depends on the objective value change caused by the addition of the best first-stage column added during that particular major iteration (i.e. the first-stage column that brought the maximum decrease in the objective at that major iteration).
  - **Comparing to the last cut/column:** If the objective change caused by the last cut added is below a certain percentage $\mathcal{P}$ of the objective change resulting from the addition of the previous cut during the same major iteration, stop searching for cuts and switch to columns. Switching back from first-stage columns to cuts, in a similar manner, depends on the objective value change caused by the addition of the first-stage column added most recently during that particular major iteration.

6.3.2.4 Stabilization Strategies Both the L-shaped method and column generation are prone to slow convergence due to degeneracy [21, 110, 177]. Hence, we also test the performances of stabilization schemes for column generation of du Merle et al. [55] and the bundle-trust region method for the L-shaped method of Linderoth and Wright [108] within the context of our combined cut/column generation framework.

Stabilization for Column Generation

Column generation methods are known for poor convergence properties. The dual variables tend to oscillate, and do not follow a smooth convergence pattern to their respective optima. Although a near-optimal solution is often approached relatively quickly, little progress is made per iteration close to the optimum [110].

In this chapter, we explore the stabilization method of du Merle et al. [55] within the context of our algorithm. This approach is a combined box-penalty and ϵ-perturbation strategy. It makes use of surplus and slack variables with penalties, which in turn is equivalent to requiring soft lower and upper bounds on the dual variables and penalizing the variables when they lie outside the bounds.

Applying stabilized first-stage column generation within the main algorithmic framework of Section 6.3.1 replaces the *First-stage Column Generation* procedure with the following, and everything else stays the same:

Stabilized First-stage Column Generation.

1. *Initialize stabilization parameters for first-stage column generation.* Let $\hat{\pi}^\nu$, $\hat{\sigma}^\nu$, $\hat{\rho}^\nu$ and $\hat{\eta}^\nu$ represent the optimal dual variables corresponding to constraint sets (6.13b), (6.13c), (6.13d) and (6.13e) at the latest iteration, respectively.

   a. Set the initial dual box:

   - $\delta^a_- = \hat{\pi}^\nu - \epsilon$, $\delta^a_+ = \hat{\pi}^\nu + \epsilon$,
   - $\delta^f_- = \hat{\sigma}^\nu - \epsilon$, $\delta^f_+ = \hat{\sigma}^\nu + \epsilon$,
   - $\delta^o_- = \hat{\rho}^\nu - \epsilon$, $\delta^o_+ = \hat{\rho}^\nu + \epsilon$, and
   - $\delta^c_- = \hat{\eta}^\nu - \epsilon$, $\delta^c_+ = \hat{\eta}^\nu + \epsilon$,

   for some $\epsilon \geq 0$. Let $\delta_- = (\delta^a_-, \delta^f_-, \delta^o_-, \delta^c_-)$ and $\delta_+ = (\delta^a_+, \delta^f_+, \delta^o_+, \delta^c_+)$. Go to step 1b.

b. Set the initial dual penalties:

- $\varepsilon_-^a = \varepsilon_0$, $\varepsilon_+^a = \varepsilon_0$,
- $\varepsilon_-^f = \varepsilon_0$, $\varepsilon_+^f = \varepsilon_0$,
- $\varepsilon_-^o = \varepsilon_0$, $\varepsilon_+^o = \varepsilon_0$, and
- $\varepsilon_-^c = \varepsilon_0$, $\varepsilon_+^c = \varepsilon_0$,

where ε_0 is the initial dual penalty parameter. Let $\varepsilon_- = (\varepsilon_-^a, \varepsilon_-^f, \varepsilon_-^o, \varepsilon_-^c)$ and $\varepsilon_+ = (\varepsilon_+^a, \varepsilon_+^f, \varepsilon_+^o, \varepsilon_+^c)$. Go to step 2.

2. *Master problem.* Set $\nu = \nu + 1$. Solve the stabilized column generation restricted master problem (RMP-ColG) below:

$$\begin{aligned}
&\min f^T z + \theta - \delta_-^a v_-^a + \delta_+^a v_+^a - \delta_-^f v_-^f + \delta_+^f v_+^f - \delta_-^o v_-^o + \delta_+^o v_+^o \quad - \delta_-^c v_-^c + \delta_+^c v_+^c \\
&\text{subject to} \quad g^T z - v_-^a + v_+^a \geq b', && && \text{(6.23a)} \\
&\bar{D}_\iota z - v_{\iota_-}^f + v_{\iota_+}^f \geq \bar{d}_\iota, && \text{for } \iota = 1, \ldots, s_f, && \text{(6.23b)} \\
&\bar{E}_\iota z + \theta - v_{\iota_-}^o + v_{\iota_+}^o \geq \bar{e}_\iota, && \text{for } \iota = 1, \ldots, s_o, && \text{(6.23c)} \\
&\sum_{i \in \tau} z_i - v_-^c + v_+^c = 1, && && \text{(6.23d)} \\
&v_-^a \leq \varepsilon_-^a, \; v_+^a \leq \varepsilon_+^a, && && \text{(6.23e)} \\
&v_-^f \leq \varepsilon_-^f, \; v_+^f \leq \varepsilon_+^f, && && \text{(6.23f)} \\
&v_-^o \leq \varepsilon_-^o, \; v_+^o \leq \varepsilon_+^o, && && \text{(6.23g)} \\
&v_-^c \leq \varepsilon_-^c, \; v_+^c \leq \varepsilon_+^c, && && \text{(6.23h)} \\
&z_i \geq 0, && \text{for } i \in \tau \cup \varphi. && \text{(6.23i)}
\end{aligned}$$

Get the optimal dual values. Let $\hat{\pi}$, $\hat{\sigma}$, $\hat{\rho}$ and $\hat{\eta}$ represent the optimal dual variables corresponding to constraint sets (6.23a), (6.23b), (6.23c) and (6.23d), respectively. Let $\tilde{\pi} = (\hat{\pi}, \hat{\sigma}, \hat{\rho}, \hat{\eta})$, $v_- = (v_-^a, v_-^f, v_-^o, v_-^c)$ and $v_+ = (v_+^a, v_+^f, v_+^o, v_+^c)$.

3. *First-stage pricing subproblem.* Solve the linear program (6.22).

- If $\hat{c} \geq \hat{\eta}$ and $v_- = v_+ = 0$, the current set of columns in the RMP is optimal. Set $NeedCols = False$. **Go to procedure *Optimality check*.**

- Otherwise,
  - if $\hat{c} < \hat{\eta}$ and is bounded, add the z column to the RMP, update τ accordingly, and set $NeedCuts = True$,
  - or, if $\hat{c} < \hat{\eta}$ and is unbounded, find the corresponding extreme ray, add the necessary z variable, update φ accordingly, and set $NeedCuts = True$.
  - *Update-δ*. If $\hat{c} \geq \hat{\eta}$, set $\delta_- = \tilde{\pi} - \epsilon$ and $\delta_+ = \tilde{\pi} + \epsilon$.
  - *Update-ε*. If $\hat{c} \geq \hat{\eta}$, set $\varepsilon_- = \frac{\varepsilon_-}{\varepsilon_{RF}}$ and $\varepsilon_+ = \frac{\varepsilon_+}{\varepsilon_{RF}}$, where ε_{RF} is the reduction factor that we use to reduce the dual penalties. Otherwise, set $\varepsilon_- = \varepsilon_- \cdot \varepsilon_{IF}$ and $\varepsilon_+ = \varepsilon_+ \cdot \varepsilon_{IF}$, , where ε_{IF} is the multiplier that we use to increase the dual penalties.
  - **Go to procedure *Optimality check*.**

Formulation (6.23) introduces surplus and slack variables, v_- and v_+, respectively, that allow for a perturbation of the right-hand side by $\varepsilon \in [-\varepsilon_+, \varepsilon_-]$, in order to help reduce degeneracy. These surplus and slack variables are penalized by δ_- and δ_+, respectively. In the dual problem, this is equivalent to penalizing the dual variables $\tilde{\pi}$ if they lie outside the box defined by $[\delta_-, \delta_+]$. The optimal solution to RMP-ColG will also be optimal for the RMP if one of the two following conditions holds: (a) $\varepsilon_- = \varepsilon_+ = 0$, or (b) $\delta_- < \tilde{\pi} < \delta_+$.

The selection of parameters $\varepsilon_-, \varepsilon_+, \delta_-$ and δ_+, and how they are updated throughout the column generation algorithm are crucial factors for the effectiveness of the algorithm. The user is flexible in defining the details of the *Update-δ* and *Update-ε* steps in the procedure above. The main idea is to increase the penalties ε_- and ε_+, and reduce the size of the box $[\delta_-, \delta_+]$ if $\tilde{\pi}$ lies in it. On the other hand, if $\tilde{\pi}$ lies outside the box, one should enlarge the box width and decrease the associated penalties. The update can actually be performed in every iteration or only if the pricing problem returns a non-negative reduced cost. The column generation procedure is guaranteed to converge finitely if the two following properties hold [55]: (a) after some number of iterations, ε_- and ε_+ decrease so that they vanish in a finite number of iterations, (b) after a given number of iterations, δ_- and δ_+ are updated only if the column returned by the pricing problem has a non-negative reduced cost.

We treat the column generation part of every major iteration as a separate column generation algorithm and use the initialization step in each major iteration as long as at least one L-shaped cut has been added from one major iteration to the other. However, if between two major iterations no cuts have been added, then we skip the initialization phase of the stabilized column generation procedure and utilize the latest values of δ and ε from the previous major iteration.

Application of this method to second-stage column generation is straightforward. Here we show how the routine for optimality-cut generation will change when stabilization is employed. If second-stage column generation is used, then Step 3 in procedure *L-shaped cut generation* should be replaced with the following:

L-shaped cut generation (*Stabilized second-stage column generation step*)

3. *Optimality cut generation.* Set the scenario index $k = 1$.

   a. *Initialize stabilization parameters for second-stage column generation.* Choose the initial dual box[1] defined by $\hat{\delta}_k$ and $\hat{\delta}_k^0$, and the dual penalty parameter ε_0.

      i. Set the initial dual box:

      - $\alpha_-^a = \hat{\delta}_k - \epsilon$, $\alpha_+^a = \hat{\delta}_k + \epsilon$, and
      - $\alpha_-^c = \hat{\delta}_k^0 - \epsilon$, $\alpha_+^c = \hat{\delta}_k^0 + \epsilon$,

      for some $\epsilon \geq 0$. Let $\alpha_- = (\alpha_-^a, \alpha_-^c)$ and $\alpha_+ = (\alpha_+^a, \alpha_+^c)$. Go to step 3aii.

      ii. Set the initial dual penalties:

      - $\varepsilon_-^a = \varepsilon_0$, $\varepsilon_+^a = \varepsilon_0$, and
      - $\varepsilon_-^c = \varepsilon_0$, $\varepsilon_+^c = \varepsilon_0$,

      where ε_0 is the initial dual penalty parameter. Let $\varepsilon_- = (\varepsilon_-^a, \varepsilon_-^c)$ and $\varepsilon_+ = (\varepsilon_+^a, \varepsilon_+^c)$. Go to step 3b.

[1] The user can either perform a minor iteration without stabilization and then start the stabilization procedure using the duals found, or store the last duals from the previous major iteration and set the initial box to those values.

b. *Optimality subproblem.* Solve the following linear program:

$$Q(z^\nu, \xi^k) = \min \sum_{i \in \bar{\tau}_k \cup \bar{\varphi}_k} \zeta_i(\xi^k) u_i - \alpha_-^a v_-^a + \alpha_+^a v_+^a - \alpha_-^c v_-^c + \alpha_+^c v_+^c \tag{6.24a}$$

$$\text{subject to} \quad \sum_{i \in \bar{\tau}_k \cup \bar{\varphi}_k} \varrho_i(\xi^k) u_i - v_-^a + v_+^a \geq h'(\xi^k) - \gamma^k z^\nu, \tag{6.24b}$$

$$\sum_{i \in \bar{\tau}_k} u_i - v_-^c + v_+^c = 1, \tag{6.24c}$$

$$v_-^a \leq \varepsilon_-^a, \; v_+^a \leq \varepsilon_+^a, \tag{6.24d}$$

$$v_-^c \leq \varepsilon_-^c, \; v_+^c \leq \varepsilon_+^c, \tag{6.24e}$$

$$u_i \geq 0, \qquad \forall i \in \bar{\tau}_k \cup \bar{\varphi}_k. \tag{6.24f}$$

Let δ_k and δ_k^0 denote the optimal dual variables corresponding to constraint sets (6.24b) and (6.24c), respectively. Let $\tilde{\delta} = (\delta_k, \delta_k^0)$, $v_- = (v_-^a, v_-^c)$ and $v_+ = (v_-+^a, v_+^c)$. If $Q(z^\nu, \xi^k)$ is unbounded, this shows that the original problem is unbounded and the algorithm terminates. Otherwise, go to step 3c (*second-stage column generation for the optimality subproblem*).

c. *Second-stage column generation for the optimality subproblem.* Solve the second-stage pricing subproblem (6.19).

- If $\bar{c}_y \geq \delta_k^0$ and $v_- = v_+ = 0$, then if $k = K$, go to step 3d (*optimality cut insertion*), otherwise set $k = k + 1$ and go to step 3a (*initialize stabilization parameters for second-stage column generation*).
- Otherwise,
  - if $\bar{c}_y = -\infty$, add a corresponding extreme ray of the polyhedron Λ_k to $\bar{\varphi}_k$, add the corresponding u variable to $Q(z, \xi^k)$,
  - or, if $-\infty < \bar{c}_y < \delta_k^0$, add the corresponding extreme point of Λ_k to $\bar{\tau}_k$, and add the corresponding u variable to the corresponding optimality subproblem.
  - *Update-α.* If $\bar{c}_y \geq \delta_k^0$, set $\alpha_- = \tilde{\delta} - \epsilon$ and $\alpha_+ = \tilde{\delta} + \epsilon$.
  - *Update-ε.* If $\bar{c}_y \geq \delta_k^0$, set $\varepsilon_- = \frac{\varepsilon_-}{\varepsilon_{RF}}$ and $\varepsilon_+ = \frac{\varepsilon_+}{\varepsilon_{RF}}$. Otherwise, set $\varepsilon_- = \varepsilon_- \cdot \varepsilon_{IF}$ and $\varepsilon_+ = \varepsilon_+ \cdot \varepsilon_{IF}$.
  - Go to step 3b (*optimality subproblem*).

d. *Optimality cut insertion.* Calculate the cut coefficients for existing variables in the RMP and the right-hand side as shown in (6.20a) and (6.20b). Calculate and store the implicit technology coefficients that will be used to construct new columns as shown in (6.21).

- If $w^\nu = \bar{e}_{s_o+1} - \bar{E}_{s_o+1} z^\nu > \theta^\nu$, add the corresponding optimality cut of the form (6.13d) and set $s_o = s_o + 1$, $NeedCols = True$. **Go to procedure *Optimality check*.**
- Otherwise, the RMP is optimal for the current set of z variables. So, no optimality cut can be added. Set $NeedCuts = False$. **Go to procedure *Optimality check*.**

Note that stabilization parameters $\alpha, \varepsilon, \varepsilon_{RF}$ and ε_{IF} may also depend on the scenario k, for $k = 1, \ldots, K$. We omitted k from our notation for these parameters for ease of exposition. This stabilization subroutine can also be applied to second-stage column generation in a very similar way.

Stabilization for L-shaped Cut Generation

We test the bundle-trust region method for the L-shaped method of Linderoth and Wright [108] within the context of our combined cut/column generation framework. This method uses a box-shaped trust region that imposes an ℓ_∞ norm bound on the size of the step. It has certain similarities to the regularized decomposition method of Ruszczyński [147] and the bundle-trust-region methods of Kiwiel [94] and Hiriart-Urruty and Lemaréchal [74], but is different in the sense that it employs linear programming subproblems and the size of the trust region is controlled directly rather than indirectly through the use of a regularization parameter.

Recall that $\mathcal{Q}(x)$, stands for the *expected recourse function* (6.3), and $Q(x, \xi^k)$ denotes the optimal second-stage objective value (6.4) for scenario ξ^k and first-stage vector x. Applying the bundle-trust-region method within the main algorithmic framework of Section 6.3.1 replaces the *L-shaped Cut Generation* procedure with the following:

Stabilized L-shaped cut generation.

1. *Initialize stabilization parameters for L-shaped cut generation.* Choose starting point $\bar{z}$. Initialize $\Delta_0, \Delta_{High}, \epsilon_{acceptZ}$ and ϵ_{tol}. Set $\Delta = \Delta_0$. Set $counter = 0$.

2. *Master problem.* Set $\nu = \nu + 1$. Solve the cut-generation restricted master problem (RMP-CutG) below:

$$\begin{aligned} \min m^{\nu}(z) = f^T z &+ \theta && (6.25a)\\ \text{subject to } (6.13b) &- (6.13e), \\ \mathbf{1}^T(-\Delta) \leq z &- \bar{z} \leq \mathbf{1}^T(\Delta), && (6.25b)\\ z_i &\geq 0, \qquad \text{for } i \in \tau \cup \varphi. && (6.25c) \end{aligned}$$

   Let (z^{ν}, θ^{ν}) denote the optimal solution to the RMP. If $s_o = 0$, then θ is assumed to be equal to $-\infty$ and is not considered in the model.

3. *Convergence test.*

   - If

$$f^T\bar{z} + \mathcal{Q}(\bar{z}) - m^{\nu}(z^{\nu}) \leq \epsilon_{tol}(1 + |f^T\bar{z} + \mathcal{Q}(\bar{z})|), \tag{6.26}$$

     then stop the cut generation iteration, set $NeedCuts = False$, and **go to procedure *Optimality check***.

   - Otherwise, go to step 4 (*function and subgradient evaluation*).

4. *Function and subgradient evaluation.*

   a. *Feasibility cut generation.* Set the scenario index $k = 1$.

      i. *Feasibility subproblem.* Solve the linear program (6.14). Let δ_k and δ_k^0 denote the optimal dual variables corresponding to constraint sets (6.14b) and (6.14c), respectively.

         - If $w' = 0$ and $k = K$, go to step 4b (*optimality cut generation*) .
         - If $w' = 0$ and $k < K$, set $k = k + 1$ and go to step 4ai (*feasibility subproblem*).

- If $w' > 0$, no feasible solution can be generated by using the current elements of the sets $\bar{\tau}_k$ and $\bar{\varphi}_k$. However, this does not show that the whole subproblem is infeasible. We have to generate more extreme points and rays for the sets $\bar{\tau}_k$ and $\bar{\varphi}_k$. Go to step 4aii (*second-stage column generation for the feasibility subproblem*).

ii. *Second-stage column generation for the feasibility subproblem.* Solve the second-stage pricing subproblem (6.15).

- If $\bar{c}_y \leq -\delta_k^0$, then no column can improve the objective function, w', so the whole second-stage subproblem is infeasible. Go to step 4aiii (*feasibility cut insertion).*
- If $\bar{c}_y = \infty$, add a corresponding extreme ray of the polyhedron Λ_k to $\bar{\varphi}_k$, add the corresponding u variable to the feasibility subproblem, and return to step 4ai (*feasibility subproblem*).
- If $-\delta_k^0 < \bar{c}_y < \infty$, add the corresponding extreme point of Λ_k to $\bar{\tau}_k$, add the corresponding u variable to the feasibility subproblem and return to step 4ai (*feasibility subproblem*).

iii. *Feasibility cut insertion.* Calculate the cut coefficients for existing variables in the RMP and the right-hand side as shown in (6.16a) and (6.16b), and add a feasibility cut of the form (6.13c). Calculate and store the *implicit technology coefficients* as shown in (6.17). Set $s_f = s_f + 1$, $NeedCols = True$. **Go to procedure *Optimality check*.**

b. *Optimality cut generation.* Set the scenario index $k = 1$.

i. *Optimality subproblem.* Solve the linear program (6.18). Let δ_k and δ_k^0 denote the optimal dual variables corresponding to constraint sets (6.18b) and (6.18c), respectively. If $Q(z^\nu, \xi^k)$ is unbounded, this shows that the original problem is unbounded and the algorithm terminates. Otherwise, go to step 4bii (*second-stage column generation for the optimality subproblem*).

ii. *Second-stage column generation for the optimality subproblem.* Solve the second-stage pricing subproblem (6.19).

- If $\bar{c}_y \geq \delta_k^0$, then if $k = K$, go to step 5 (*trust-region center test*), otherwise set $k = k + 1$ and go to step 4bi (*optimality subproblem*) .
- If $\bar{c}_y = -\infty$, add a corresponding extreme ray of the polyhedron Λ_k to $\bar{\varphi}_k$, add the corresponding u variable to $Q(z, \xi^k)$, and return to step 4bi (*optimality subproblem*).
- If $-\infty < \bar{c}_y < \delta_k^0$, add the corresponding extreme point of Λ_k to $\bar{\tau}_k$, add the corresponding u variable to the corresponding optimality subproblem and return to step 4bi (*optimality subproblem*).

5. *Trust-region center test.*
   - If
   $$f^T z^\nu + \mathcal{Q}(z^\nu) \leq (f^T \bar{z} + \mathcal{Q}(\bar{z})) - \epsilon_{acceptZ}(f^T \bar{z} + \mathcal{Q}(\bar{z}) - m^\nu(z^\nu)), \tag{6.27}$$
   – *Re-center the trust-region.* Set $\bar{z} = z^\nu$.
   – *Increase trust-region radius.* If
   $$f^T z^\nu + \mathcal{Q}(z^\nu) \leq (f^T \bar{z} + \mathcal{Q}(\bar{z})) - 0.5(f^T \bar{z} + \mathcal{Q}(\bar{z}) - m^\nu(z^\nu)), \quad \|\bar{z} - z^\nu\|_\infty = \Delta, \tag{6.28}$$
   set $\Delta = \min(\Delta_{High}, 2\Delta)$, and go to step 2 (*master problem*).
   - Otherwise, go to step 6 (*optimality cut insertion*).
6. *Optimality cut insertion.* Calculate the cut coefficients for existing variables in the RMP and the right-hand side as shown in (6.20a) and (6.20b). Calculate and store the implicit technology coefficients that will be used to construct new columns as shown in (6.21).
   - If $w^\nu = \bar{e}_{s_0+1} - \bar{E}_{s_0+1} z^\nu > \theta^\nu$, add the corresponding optimality cut of the form (6.13d) and set $s_o = s_o + 1$, $NeedCols = True$. Go to step 7 (*reduce-*Δ).
   - Otherwise, the RMP is optimal for the current set of z variables. So, no optimality cut can be added. Set $NeedCuts = False$. **Go to procedure *Optimality check***.
7. *Reduce-*Δ. Evaluate
   $$\rho = \min(1, \Delta) \frac{(f^T z^\nu + \mathcal{Q}(z^\nu)) - (f^T \bar{z} + \mathcal{Q}(\bar{z}))}{(f^T \bar{z} + \mathcal{Q}(\bar{z})) - m^\nu(z^\nu)}, \tag{6.29}$$
   - if $\rho > 0$, set $counter = counter + 1$,

- if $\rho > 3$ or ($counter \geq 3$ and $\rho \in (1, 3]$),
  - set $\Delta = \frac{1}{\min(\rho,4)}\Delta$,
  - reset $counter = 0$.
- **Go to procedure *Optimality check***.

At the beginning of each stabilized L-shaped cut generation subroutine in every major iteration, we set $\bar{z}$ to the first-stage solution found during the last iteration of column generation, and set $\Delta = \Delta_0$, if at least one column has been added since the previous major iteration. If this is not the case, then we skip the initialization phase of the above procedure and use the latest values of $\bar{z}$ and Δ from the previous major iteration.

6.4 COMPUTATIONAL RESULTS

As mentioned before, this chapter aims to solve stochastic programs whose natural formulation involves a large number of columns efficiently through integrating column generation within the L-shaped method. To illustrate this idea of combined column and cut generation, and to evaluate the algorithmic variants defined in Section 6.3.2.3, we introduce two-stage stochastic versions of the well-known cutting stock and multi-commodity flow problems. We used a UNIX machine with AMD Opteron 240 processor and 3.8 GB RAM for our computational tests, and coded the optimization algorithms using C++ and the CPLEX 9.0 Callable Library.

6.4.1 A Two-Stage Stochastic Cutting Stock Problem (SCSP)

The classical (deterministic) cutting stock problem is a widely studied application of column generation. In the deterministic version of the cutting stock problem, we have a number of rolls of paper (or steel, etc.) of fixed width waiting to be cut. Different customers demand different numbers of rolls of various-sized widths. The problem is how to cut the rolls so as to minimize the number of rolls used.

The deterministic cutting stock problem was introduced by Kantorovich [89] and a column generation formulation was presented by Gilmore and Gomory [70]. Valério de Carvalho and Rodrigues [172] solved a two-stage cutting stock problem. In their definition of the two-stage cutting stock problem, the stock was cut twice, which is usually the case in the cutting stock literature, but they ignored uncertainty. Vanderbeck [176] combined column and cut generation in a branch-and-price-and-cut algorithm that minimized the number of setups. Super-additive cuts were applied within the branch-and-bound tree to improve computational efficiency. Krichagina et al. [98] introduced the two-stage stochastic cutting stock problem with uncertain demand and used a suboptimal two-step approach to solve it. The first step was a linear programming problem, and the second step used Brownian control. See Ben Amor and Valério de Carvalho [16] for a recent survey on cutting stock problems.

In our two-stage stochastic cutting stock problem, we have two different sets of customer orders: a set of deterministic orders for which we know the amounts demanded and widths, and a stochastic set of orders for which widths are known but the amounts demanded vary among scenarios. Failure to meet demands or surplus products leads to shortage and surplus costs. The shortage costs may represent the cost of missing customer orders while the surplus cost may correspond to a storage or disposal cost. The stock is only cut once, and the second-stage variables only show the shortage/surplus amounts. Thus, the problem is how to cut the rolls so as to minimize expected wasted leftovers and shortage/surplus penalties.

In our model, let $\mathcal{M}_1$ be the set of deterministic orders, $\mathcal{M}_2$ be the set of variable orders, $\mathcal{N}$ denote the set of possible cutting patterns, and $\xi^1, \ldots, \xi^K$ be a set of scenarios, where scenario ξ^k occurs with probability p^k.

First-Stage Model

- $z \in \mathbb{R}_+^{|\mathcal{N}|}$ is the vector showing the number of times patterns are cut.
- $b \in \mathbb{R}_+^{|\mathcal{M}_1|}$ is the demand vector for deterministic orders.
- $A \in \mathbb{R}_+^{|\mathcal{M}_1| \times |\mathcal{N}|}$ is the deterministic cutting matrix such that each entry, (a_{ij}), shows how many units of the deterministic product i is cut when a single pattern j is used.
- $\mathbf{1} \in \mathbb{R}^{|\mathcal{N}|}$ is a vector with all entries equal to 1.

In this formulation, using a pattern corresponds to using a full roll, and a pattern may consist of both deterministic and stochastic orders.

Then the first-stage model can be written as follows::

$$\min \ \mathbf{1}^T z + \mathrm{E}_{\tilde{\xi}} Q(z, \xi^k) \tag{6.30a}$$

$$\text{subject to} \ \ Az \geq b, \tag{6.30b}$$

$$z \geq 0. \tag{6.30c}$$

Second-Stage Model

- $\Gamma \in \mathbb{R}_+^{|\mathcal{M}_2| \times |\mathcal{N}|}$ is the second-stage cutting matrix where each entry, (γ_{ij}), shows the amount of the stochastic product i cut using a single pattern j.
- $d_s(\xi^k)$, $d_e(\xi^k) \in \mathbb{R}_+^{|\mathcal{M}_2|}$ are shortage and surplus cost vectors for stochastic orders under scenario ξ^k, respectively.
- $y_s(\xi^k)$, $y_e(\xi^k) \in \mathbb{R}_+^{|\mathcal{M}_2|}$ show the amounts of shortage and surplus for stochastic orders under scenario ξ^k, respectively.
- $h(\xi^k) \in \mathbb{R}_+^{|\mathcal{M}_2|}$ is the demand vector for stochastic orders under scenario ξ^k.

Then, for $k = 1, \ldots, K$,

$$Q(z, \xi^k) = \min d_s(\xi^k)^T y_s(\xi^k) + d_e(\xi^k)^T y_e(\xi^k) \tag{6.31a}$$

$$\text{subject to } \Gamma z + y_s(\xi^k) - y_e(\xi^k) = h(\xi^k), \tag{6.31b}$$

$$y_s(\xi^k) \geq 0, \tag{6.31c}$$

$$y_e(\xi^k) \geq 0. \tag{6.31d}$$

First-Stage Pricing Subproblem

- $x \in \mathbb{R}_+^{|\mathcal{M}_1|+|\mathcal{M}_2|}$ is the vector showing the amounts of products cut by using the pattern.
- $W \in \mathbb{R}_+$ is the total width of the roll to be cut.
- $w \in \mathbb{R}_+^{|\mathcal{M}_1|+|\mathcal{M}_2|}$ is the vector of product widths.
- $\pi \in \mathbb{R}^{|\mathcal{M}_1|}$ is the dual vector corresponding to the original problem constraints in the RMP.

- $\rho \in \mathbb{R}_+^{s_o}$ is the dual vector corresponding to the current optimality cuts in the RMP, where s_o is the number of optimality cuts in the RMP at some iteration where the pricing problem is solved.

  Then the first-stage pricing subproblem can be written as a knapsack problem as follows:

$$\begin{aligned} \min \bar{c}_z = 1 - \sum_{i \in \mathcal{M}_1} \pi_i x_i - \sum_{i \in \mathcal{M}_2} (\sum_{\iota=1}^{s_o} \rho_\iota \tilde{E}_\iota^i) x_i \\ \text{subject to} \sum_{i \in \mathcal{M}_1 \cup \mathcal{M}_2} w_i x_i \leq W, \\ x \in \mathbb{Z}_+^{|\mathcal{M}_1|+|\mathcal{M}_2|}. \end{aligned}$$

where $\tilde{E}_\iota^i$ corresponds to the i^{th} column of $\tilde{E}_\iota$, $\iota = 1, \ldots, s_o$ (see (6.21)). If $\bar{c}_z \geq 0$, there is no need to add any new patterns to the RMP for the given set of optimality cuts and original problem constraints. Otherwise, if $\bar{c}_z < 0$, add the z column corresponding to this pattern to the RMP.

Note that, normally, we should also require that the z variables be integer values. However, in this chapter we used the linear relaxation of the SCSP to demonstrate the computational performances of different algorithmic approaches to the column generation model that we propose. In this particular problem, all first-stage solutions have feasible second-stage solutions. So, the "feasibility cut" step of the general algorithm disappears in the solution procedure for the SCSP. Also, note that the SCSP formulation has a simple recourse structure, which would actually ease the solution procedure, if properly exploited. However, during the computational experiments, we did not exploit this structure and treated the problem as a general two-stage stochastic linear model.

As the initial feasible columns, we used x^i, where the i^{th} component of x^i is $\left\lfloor \frac{W}{w_i} \right\rfloor$ and its remaining components are 0, for $i = 1, \ldots, \mathcal{M}_1 + \mathcal{M}_2$.

6.4.1.1 Computational Results for SCSP

We tested the different algorithmic strategies mentioned in Section 6.3.2.3 on five problem classes and generated 10 instances for each class. These problem classes are based on real-world deterministic instances provided by Vanderbeck [175]. These deterministic instances were modified to include some uncertainty

Table 6.1: Characteristics of the instances tested for SCSP.

Class Name	Deterministic Orders	Stochastic Orders	Number of Patterns	Number of Maximal Patterns	Scenarios	Instances
CSTR18p22	9	9	2,341,570	1,222,875	2500	10
CSTR25p0	12	13	4,090,522	2,374,200	1500	10
CSTR28p0	14	14	251,561,215	126,412,627	1500	10
CSTR30p0	15	15	76,713,823	47,845,631	1500	10
CSTRd43p21	21	22	1,397,337	612,034	1500	10

in the following manner: at least half of the orders were set to be stochastic and their corresponding demand and cost parameters were pulled from normal distributions where mean μ equals the value given in the deterministic case and standard deviation $\sigma = \rho\mu$ where $\rho \in \{0.1, 0.2, 0.25\}$. Table 6.1 summarizes the characteristics of the problem classes used throughout the computational experiments. This table also displays information on the number of all possible (and maximal) cutting patterns, i.e., columns, for each of the problem classes. These numbers suggest that an explicit enumeration of cutting patterns would be impractical.

In this section we summarize our computational results. We group our experiments with respect to the specific algorithmic strategies tested in each group. Each subsection starts with information on the notation for strategies and parameter settings that we tested in order to make it easier for the reader to follow the tables.

Tests on Switching Criteria and L-shaped Cut Aggregation Strategies

Let $\mathcal{L}$ denote the number of scenarios that are grouped in a scenario set for the aggregate L-shaped cut approach, that is, dual information from $\mathcal{L}$ scenarios is aggregated into one cut for each scenario set. We tested for $\mathcal{L} \in \{1, 50, 100, 500, K\}$ where $K = 1500$ for all problem classes except CSTR18p22, for which $K = 2500$. Note that $\mathcal{L} = 1$ is the multi-cut L-shaped approach, and $\mathcal{L} = K$ is the single-cut L-shaped method.

We tested the following switching criteria: *AllCutsAllCols* (All-cuts-all-columns), *AllCutsOneCol* (All-cuts-one-column), *OneCutAllCols* (One-cut-all-columns), *FirstP* (Comparing to the first cut/column), *BestP* (Comparing to the best cut/column), and *LastP* (Comparing to the last cut/column). Explanations of these can be found in Section 6.3.2.3.

We first solved our test problems using column generation on the extensive forms of the stochastic programs. These results formed a base case for our computational tests. Table 6.2 shows the results on the performances of the switching strategies *AllCutsAllCols*, *AllCutsOneCol* and *OneCutAllCols*, and compares them with column generation on the extensive form. For each one of these switching criteria, we also tested different L-shaped cut aggregation strategies. Table 6.2 shows that cut aggregation strategies are very effective, and that *AllCutsAllCols* seemed to be the best strategy among the ones tested initially. We decided to drop column generation on the extensive form, *AllCutsOneCol* and *OneCutAllCols* from further tests.

Next, we focused on threshold switching strategies and examined the affect of cut aggregation on these classes of algorithms. For threshold approaches (i.e., *FirstP*, *BestP* and *LastP*), we set the *switching threshold* $\mathcal{P}$, which is the relative tolerance between the objective change of the most recent cut/column compared to that of an earlier cut/column, to 0.25, 0.50, 0.75, 1.0 and 2.0. Our initial tests suggested that decreasing $\mathcal{P}$ along the way was an effective strategy, so we also introduced a modification factor, namely $\mathcal{P}_{factor}$, to adjust the threshold limit between major iterations, and tested for $\mathcal{P}_{factor} \in \{0.25, 0.50, 0.75, 1.0, 1.5, 2.0\}$. Hence, if $\mathcal{P}^l$ denotes the threshold limit in major iteration l, then in the next major iteration we set $\mathcal{P}^{l+1} = \mathcal{P}^l \cdot \mathcal{P}_{factor}$.

Table 6.3 compares the performances of our threshold strategies with respect to different cut-aggregation and switching tolerance schemes. For each value of $\mathcal{L}$, we display the pa-

Table 6.2: Summary of initial computational runs for SCSP (averages over all 50 instances).

Algorithm Class	$\mathcal{L}$	# of iterations		# of		CPU time
		Major	Minor	Columns	Cuts	(sec.)
CGonExtensiveForm	-	94.5	-	93.5	-	69.4
AllCutsAllCols	1	2.2	86.2	80.4	3294.4	85.2
	50	2.2	94.0	80.6	317.0	28.4
	100	2.0	101.8	83.8	236.0	32.4
	500	2.2	146.6	87.0	173.4	39.6
	K	4.2	243.0	84.2	154.8	69.1
AllCutsOneCol	1	41.2	123.4	79.8	3508.8	92.7
	50	44.0	139.6	85.2	332.4	42.0
	100	43.6	143.0	84.4	246.8	41.5
	500	44.2	187.6	86.0	179.8	55.7
	K	42.0	286.2	81.4	161.8	79.9
OneCutAllCols	1	3.6	114.1	106.9	3938.0	140.9
	50	4.7	125.8	116.5	156.0	48.71
	100	5.5	152.6	141.7	92.3	52.4
	500	9.9	250.6	230.9	32.2	76.5
	K	20.0	431.6	391.6	23.8	126.0

Table 6.3: Summary of tests on switching criteria and cut-aggregation strategies for SCSP (averages over all 50 instances).

Algorithm Class	Parameters			# of iterations		# of		CPU time
	$\mathcal{L}$	$\mathcal{P}$	$\mathcal{P}_{factor}$	Major	Minor	Columns	Cuts	(sec.)
	1	-	-	2.2	86.2	80.4	3294.4	85.2
	50	-	-	2.2	94.0	80.6	317.0	28.4
$AllCutsAllCols$	100	-	-	2.0	101.8	83.8	236.0	32.4
	500	-	-	2.2	146.6	87.0	173.4	39.6
	K	-	-	4.2	243.0	84.2	154.8	69.1
	1	.75	.50	9.0	95.2	83.6	3436.6	38.7
	50	.50	.25	6.8	98.5	83.8	306.2	10.3
$FirstP$	100	.25	.50	7.0	101.0	81.4	231.0	11.6
	500	.75	.25	7.2	140.4	82.4	164.8	18.1
	K	2.0	1.0	84.6	169.2	83.6	84.6	32.6
	1	.50	.25	7.2	88.6	78.8	3440.2	36.4
	50	.50	.25	7.9	97.9	82.2	308.6	10.4
$BestP$	100	.50	.75	12.8	105.0	81.2	223.6	10.9
	500	.25	.25	6.6	140.0	80.4	171.2	17.0
	K	1.0	1.0	33.6	178.4	85.2	91.8	33.1
	1	.50	.25	6.0	89.6	80.8	3420.8	32.8
	50	.50	.25	6.3	95.3	80.1	310.5	9.8
$LastP$	100	.50	.25	6.2	103.6	83.2	234.2	9.9
	500	2.0	.50	9.6	136.2	79.0	160.2	16.0
	K	.75	1.0	32.4	184.0	86.6	94.6	30.0

rameter setting for $\mathcal{P}$ and $\mathcal{P}_{factor}$ that had the fastest average solution time. Table 6.3 shows that algorithms using threshold switching criteria and cut-aggregation clearly dominate the *AllCutsAllCols* approach. After examining these results, *AllCutsAllCols* was excluded from further testing, and the standard parameter settings for all threshold approaches were set to $\mathcal{P} = 0.50$ and $\mathcal{P}_{factor} = 0.25$ for the remainder of the testing process.

Tests on Column Generation Strategies

We tested the following pricing strategies: *OPri* (solving the pricing problem to optimality and adding the column of the optimal solution), *MPri* (solving the pricing problem to optimality and adding all favorable column found on the way), and *FPri* (solving the pricing problem only until a favorable column is found without fully optimizing it).

Table 6.4 compares the performances of our column generation strategies. These results show that multiple pricing could be an effective strategy for the design of algorithms. However, *FPri* did not appear to be competitive and was eliminated from further testing.

Tests on Stabilization Strategies

For stabilized column generation, we tested for $\varepsilon_0 \in \{0.001, 0.01, 0.1, 1.0\}$, $\varepsilon_{RF} \in \{10, 100, 1000\}$, and $\varepsilon_{IF} \in \{1.001, 1.01, 1.1, 1.2, 1.5, 2.0\}$. For stabilized L-shaped cut generation, we tested for $\Delta_0 \in \{1, 10, 100, 1000\}$, $\Delta_{High} \in \{10, 100, 1000, 10000\}$, and $\epsilon_{acceptZ} \in \{0.0001, 0.001, 0.01, 0.1\}$. For a combined stabilization approach for both column and cut generation, we tested combinations of these parameters together.

Table 6.5 shows the results for stabilized algorithms when compared to algorithms without any stabilization schemes. We tested for threshold switching strategies, and both multiple and optimal pricing options. Results for stabilized algorithms are results obtained with parameter settings that had the overall fastest average run time.

The fact that stabilization did not appear to help solution times for our test set motivated us to generate larger instances to assess the performance of stabilization on bigger problems. We generated 50 instances of the problem class CSTRd43p21, each with 10000 scenarios. After running some initial tests for cut aggregation, we set $\mathcal{L} = 500, \mathcal{P} = .50, \mathcal{P}_{factor} = .25$. Table 6.6 summarizes the results for algorithms with different threshold switching, pricing

Table 6.4: Summary of computational runs for testing pricing strategies (with settings $\mathcal{L} = 50, \mathcal{P} = .50, \mathcal{P}_{factor} = .25$).

Algorithm		# of iterations		# of		CPU time
Switching	Pricing	Major	Minor	Columns	Cuts	(sec.)
	$OPri$	6.8	98.5	83.8	306.2	10.3
$FirstP$	$MPri$	6.0	70.2	136.2	299.4	10.2
	$FPri$	11.7	171.2	151.7	304.7	16.3
	$OPri$	7.9	97.9	82.2	308.2	10.4
$BestP$	$MPri$	6.7	69.2	132.7	300.7	10.2
	$FPri$	13.8	173.7	152.1	309.5	18.0
	$OPri$	6.3	95.3	80.1	310.5	9.8
$LastP$	$MPri$	5.3	69.4	134.5	309.1	9.5
	$FPri$	12.4	172.1	151.0	307.9	16.8

and stabilization strategies. The table is sorted in ascending order of average solution times, and also displays information on the number of times an algorithm was the fastest to solve a problem instance and the number of times it was in the top 5.

Results from Tables 6.5 and 6.6 suggest that performance gains from stabilization become more apparent on larger problems. For comparison purposes we also solved the same instances of Table 6.6 using column generation on the extensive form. The average solution time for this approach was 1634 seconds.

Tables 6.2 through 6.6 also show that our algorithms end up generating only a small fraction of the total number of potential columns, which can be seen on Table 6.1.

6.4.2 A Two-Stage Stochastic Multi-commodity Flow Problem (SMCFP)

Distribution problems are naturally modeled on a network, with commodities flowing between nodes along arcs. Each node has a net supply (with negative numbers corresponding to demand), and each arc has a capacity. In many applications, multiple commodities must be routed through a network. In such a model, each node has a vector of net supplies, and the total amount traversing an arc must respect the capacity restrictions [3]. There are two main approaches to solving multi-commodity network flow problems. The *arc-based* formulation has variables that specify the amount of each commodity that flows along each arc. Flow-balance constraints are separable by commodity, but arc capacity constraints are not. These models are typically solved using Lagrangian relaxation [3, 128]. The *path-based* formulation defines a variable for every commodity, and every possible path from a source to a sink. The variable describes how much should flow along each path. There are typically far more paths than could be explicitly stated, so a column generation approach based on Dantzig-Wolfe decomposition [42] is employed. This generates new paths dynamically based on the dual solution to a restricted master problem.

Table 6.5: Summary of tests on pricing and stabilization strategies (with settings $\mathcal{L} = 50, \mathcal{P} = .50, \mathcal{P}_{factor} = .25$).

Algorithm			# of iterations		# of		CPU time
Switching	Pricing	Stabilization	Major	Minor	Columns	Cuts	(sec.)
FirstP	*OPri*	-	6.8	98.5	83.8	306.2	10.3
		StabilizedColGen [a]	6.3	97.6	81.6	307.5	10.2
		StabilizedCutGen [b]	9.2	115.2	97.5	307.5	12.1
		StabilizedCol&CutGen [c]	9.0	121.5	96.9	302.2	12.5
	MPri	-	6.0	70.2	136.2	299.4	10.2
		StabilizedColGen	5.8	68.1	132.4	301.4	9.9
		StabilizedCutGen	9.5	89.1	169.0	311.8	14.7
		StabilizedCol&CutGen	9.4	98.0	168.8	310.6	15.2
BestP	*OPri*	-	7.9	97.9	82.2	308.6	10.4
		StabilizedColGen	7.5	100.8	84.6	309.0	10.7
		StabilizedCutGen	7.6	104.8	88.2	312.9	11.2
		StabilizedCol&CutGen	7.6	107.1	88.0	312.6	11.5
	MPri	-	6.7	69.2	132.7	300.7	10.2
		StabilizedColGen	6.5	69.5	130.5	300.0	10.1
		StabilizedCutGen	7.0	78.0	148.9	309.7	11.5
		StabilizedCol&CutGen	6.9	79.1	144.3	310.0	11.7
LastP	*OPri*	-	6.3	95.3	80.1	310.5	9.8
		StabilizedColGen	6.0	96.9	81.2	311.6	9.9
		StabilizedCutGen	6.7	104.5	88.7	313.3	10.8
		StabilizedCol&CutGen	6.4	103.5	86.8	312.3	10.7
	MPri	-	5.3	69.4	134.5	309.1	9.5
		StabilizedColGen	5.1	70.3	131.2	309.3	9.6
		StabilizedCutGen	5.8	74.9	141.4	309.0	10.5
		StabilizedCol&CutGen	5.9	76.8	140.2	309.7	10.9

[a] Stabilized column generation with $\varepsilon_0 = 0.001, \varepsilon_{RF} = 100, \varepsilon_{IF} = 1.1$.

[b] Stabilized cut generation with $\Delta_0 = 100, \Delta_{High} = 1000, \epsilon_{acceptZ} = 0.001$.

[c] Stabilized column and cut generation with $\varepsilon_0 = 0.001, \varepsilon_{RF} = 1000, \varepsilon_{IF} = 1.1; \Delta_0 = 10, \Delta_{High} = 100, \epsilon_{acceptZ} = 0.01$.

Table 6.6: Summary of tests on larger instances with 10,000 scenarios for SCSP (with settings $\mathcal{L} = 500, \mathcal{P} = .50, \mathcal{P}_{factor} = .25$).

Algorithm			Iterations		# of		CPU time	# of times	
Switching	Pricing	Stabilization	Maj	Min	Col	Cut	(sec.)	Top 5	Fastest
LastP	*OPri*	*StabilizedColGen*	6.4	134.3	119.0	187.2	81.6	27	6
FirstP	*OPri*	*StabilizedCol&CutGen*	8.3	154.1	133.3	185.2	82.6	10	0
FirstP	*OPri*	*StabilizedCutGen*	8.1	148.9	133.1	186.0	83.3	15	1
LastP	*MPri*	-	5.7	91.0	192.5	186.4	85.1	16	6
FirstP	*MPri*	-	6.3	91.1	191.7	177.8	85.2	20	7
BestP	*OPri*	*StabilizedColGen*	8.1	134.5	118.5	187.4	85.2	14	0
LastP	*MPri*	*StabilizedColGen*	5.7	92.3	191.3	186.2	88.5	18	6
FirstP	*OPri*	*StabilizedColGen*	7.0	133.2	118.1	181.3	90.0	19	9
LastP	*OPri*	*StabilizedCutGen*	7.7	142.5	125.9	194.8	90.2	10	0
LastP	*OPri*	*StabilizedCol&CutGen*	7.3	145.2	126.7	195.3	92.1	13	2
BestP	*MPri*	*StabilizedColGen*	7.1	92.4	189.7	183.0	93.9	7	1
FirstP	*MPri*	*StabilizedColGen*	6.0	91.1	188.4	178.1	96.1	12	2
FirstP	*OPri*	-	8.1	134.4	118.9	181.7	96.2	14	2
BestP	*MPri*	-	7.1	90.7	189.5	183.5	96.6	13	2
BestP	*OPri*	*StabilizedCutGen*	8.9	144.8	127.0	196.5	100.7	5	0
LastP	*OPri*	-	7.6	135.6	119.6	188.7	101.8	17	4
LastP	*MPri*	*StabilizedCol&CutGen*	6.2	100.5	202.4	194.8	104.6	1	0
LastP	*MPri*	*StabilizedCutGen*	6.0	97.6	198.8	195.8	104.7	8	1
BestP	*OPri*	*StabilizedCol&CutGen*	8.9	150.3	130.1	196.4	105.6	4	0
BestP	*MPri*	*StabilizedCol&CutGen*	7.4	104.5	207.0	195.5	110.2	1	0
BestP	*OPri*	-	8.9	133.9	117.7	187.4	115.3	4	1
BestP	*MPri*	*StabilizedCutGen*	7.4	101.1	204.3	196.5	115.8	2	0
FirstP	*MPri*	*StabilizedCol&CutGen*	8.2	118.0	239.6	187.4	117.5	0	0
FirstP	*MPri*	*StabilizedCutGen*	8.2	110.2	240.1	187.5	121.2	0	0

In the proposed framework, the net supply vectors and arc capacities are unknown in the first stage. The decision maker has an opportunity to "pre-position" commodities at various nodes before the uncertainty is realized. As soon as the uncertainty has been realized, the net node supplies and arc capacities are known. The objective is to minimize the cost of pre-positioning the commodities plus the expected cost of meeting the demand when the uncertainty is realized.

Deterministic multi-commodity flow problems are well studied; for a survey see [3]. Recent literature on the integer multi-commodity flow problem includes [13, 29, 93]. Although several multi-commodity flow models have combined column and cut generation [13, 93], the cuts are valid inequalities that improve computational efficiency. The only paper on stochastic multi-commodity flow of which we are aware is [71]. They solved an arc-based multi-commodity flow model with random arc capacities. In this chapter we consider a model more general than the one they developed.

We assume that the second-stage capacity is unaffected by the amount of first-stage flow, but the model can be easily modified to handle such a case. Let

- $G = (N, \mathcal{A})$ be a directed graph.
- L be a set of commodities.
- $\xi^1, \ldots, \xi^K$ be a set of scenarios, where scenario ξ^k occurs with probability p^k.
- u_{ij}^0 be the first-stage capacity for arc $(i, j) \in \mathcal{A}$.
- z be the *pre-position vector*, so that z_i^l units of commodity l have been pre-positioned at node i in the first stage. A negative value of z implies the corresponding commodity should be moved from the node.
- $g_i^l \geq 0$ be the initial supply of commodity $l \in L$ at node $i \in N$.

First-Stage Model:

- Let Π be the set of all simple directed paths.
- For each node $i \in N$, let Π_i^+ (Π_i^-) be the set of paths to (from) node i.
- For each path P, let $\delta_{ij}^P = 1$ if path P uses arc $(i, j) \in \mathcal{A}$, and 0 otherwise.
- For each commodity $l \in L$, and each path $P \in \Pi$, let f_P^l be the amount of commodity l that flows along path P.

- Let c_P^l be the cost of sending a single unit of commodity l along path P.

Then the path-based first-stage model is as follows:

$$\min \sum_{P\in\Pi}\sum_{l\in L} c_P^l f_P^l \;+\; \mathrm{E}_{\tilde{\xi}} Q(z,\xi^k))$$

$$\text{subject to} \sum_{P\in\Pi_i^+} f_P^l - \sum_{P\in\Pi_i^-} f_P^l \;=\; z_i^l, \qquad \forall\, i\in N,\ \forall\, l\in L, \tag{6.32a}$$

$$\sum_{P\in\Pi}\sum_{l\in L} f_P^l \delta_{ij}^P \;\le\; u_{ij}^0, \qquad \forall (i,j)\in\mathcal{A}, \tag{6.32b}$$

$$f_P^l \;\ge\; 0, \qquad \forall\, l\in L,\ \forall P\in\Pi, \tag{6.32c}$$

$$z_i^l \;\ge\; -\max(g_i^l, 0), \qquad \forall\, i\in N,\ \forall\, l\in L. \tag{6.32d}$$

The lower bound on z, (6.32d), means that a node may not send out more of a commodity than its initial supply.

Second-Stage Model:

Define:

- $h_i^l(\xi^k)$ to be the net demand of commodity l at node i under scenario ξ^k.
- For each arc $(i,j)\in\mathcal{A}$, and each scenario ξ^k, let $u_{ij}(\xi^k)$ be its second-stage capacity under scenario ξ^k.

  For each scenario define:

- $f_P^l(\xi^k)$ to be the amount of commodity l that flows along path $P\in\Pi$ for any commodity $l\in L$ under scenario ξ^k.
- $c_P^l(\xi^k)$ to be the cost of sending a single unit of commodity l along path P under scenario ξ^k.

Then for each scenario ξ^k,

$$
\begin{aligned}
Q(z,\xi^k) \;=\; & \min \sum_{l\in L}\sum_{P\in\Pi} c_P^l(\xi^k) f_P^l(\xi^k) \\
\text{subject to } & \sum_{P\in\Pi_i^+} f_P^l(\xi^k) - \sum_{P\in\Pi_i^-} f_P^l(\xi^k) \;=\; h_i^l(\xi^k) \;-\; z_i^l, \quad \forall\, i\in N,\ \forall\, l\in L, && \text{(6.33a)}\\
& \sum_{l\in L}\sum_{P\in\Pi} f_P^l(\xi^k)\delta_{ij}^P \;\le\; u_{ij}(\xi^k), \qquad \forall (i,j)\in\mathcal{A}, && \text{(6.33b)}\\
& f_P^l(\xi^k) \;\ge\; 0, \qquad \forall\, l\in L,\ \forall P\in\Pi_l^i(\xi^k). && \text{(6.33c)}
\end{aligned}
$$

First-Stage Pricing Problems:

Assume we add two dummy nodes to the underlying network: a super-supply and a super-demand node. Let nodes 0 and S denote the super-supply and super-demand nodes, respectively. There will be an arc going from 0 to each node in the network, except S. Similarly, there will be an arc going from each node in the network, except 0, to S. A first-stage pricing subproblem will be used to generate a "favorable" path, namely P, from 0 to S in order to perform the "pre-positioning" operations. Now, let

- $x_{ij}^l = \begin{cases} 1, \text{ if the arc } (i,j) \text{ is in path P,} \\ 0, \text{ otherwise} \end{cases} \quad (i,j)\in\mathcal{A}, l\in L.$
- c_{ij}^l, $(i,j)\in\mathcal{A}, l\in L$, denote the cost of sending one unit of commodity l from i to j.
- σ_i^l, $i\in N, l\in L$, denote the dual variables corresponding to constraints (6.32a).
- w_{ij}, $(i,j)\in\mathcal{A}$, denote the dual variables corresponding to constraints (6.32b).
- $\delta^+(i)$, $i\in N$, denote the set of nodes $j\in N$ such that $(i,j)\in\mathcal{A}$.
- $\delta^-(i)$, $i\in N$, denote the set of nodes $j\in N$ such that $(j,i)\in\mathcal{A}$.

Then, the first-stage pricing subproblem for commodity $l\in L$ can be formulated as:

$$\min \bar{c}_P^l = \sum_{(i,j)\in\mathcal{A}} (c_{ij}^l - w_{ij}) x_{ij}^l + \sum_{i\in N} \sigma_i^l x_{0i}^l - \sum_{i\in N} \sigma_i^l x_{iS}^l$$

subject to

$$\sum_{j\in\delta^+(i)} x_{ij}^l - \sum_{j\in\delta^-(i)} x_{ij}^l = \begin{cases} 1, & \text{if } i=0 \\ 0, & \text{if } i\in N, i\neq 0 \text{ and } i\neq S \\ -1, & \text{if } i=S \end{cases}$$

$$x_{ij}^l \geq 0, \quad (i,j) \in \mathcal{A}.$$

If $\bar{c}_P^l \geq 0$, there is no need to add the path P to the second-stage subproblem corresponding to scenario ξ^k. Otherwise, if $\bar{c}_P^l < 0$, path P should be added. Note that we just have a shortest path problem with modified arc costs as the pricing subproblem for each scenario and commodity.

Second-Stage Pricing Problems:

The second-stage pricing subproblems will be used to generate a "favorable" path from 0 to S for satisfying the demand for commodity l under scenario ξ^k, $l \in L, k = 1, \ldots, K$. We have one second-stage pricing problem for each commodity $l \in L$ and scenario $\xi^k \in \Xi$, and their parameters and formulations will be the same as their first-stage counterparts, except for the fact that σ and w denote to the dual variables corresponding to the second-stage constraints (6.33a) and (6.33b), respectively, in this case.

6.4.2.1 Computational Results for SMCFP In this section we summarize our computational results our stochastic multi-commodity flow problems. We use the same notation developed in earlier sections of the chapter. We tested the different algorithmic strategies mentioned in Section 6.3.2.3 on four problem classes and generated 10 instances for each class. These problem classes are based on real-world deterministic instances provided by Vance [123]. These deterministic instances were modified to include some uncertainty in the following manner: node demand/supply and arc capacity parameters were pulled from normal distributions where mean μ equals the value given in the deterministic case and standard deviation $\sigma = \rho\mu$ where $\rho \in \{0.1, 0.25, 0.50\}$. Table 6.7 summarizes the characteristics of the problem classes used throughout the computational experiments.

For cut-aggregation strategies, we tested for $\mathcal{L} \in \{1, 5, 10, 25, 50\}$. Note that $\mathcal{L} = 1$ is the multi-cut L-shaped approach, and $\mathcal{L} = 50$ is the single-cut L-shaped method. Table 6.8 shows the results on the performances of the switching strategies *AllCutsAllCols*, *AllCutsOneCol* and *OneCutAllCols*. For each one of these switching criteria, we also tested different L-shaped cut aggregation strategies. Table 6.8 shows that cut aggregation strategies are very effective, and that *AllCutsAllCols* seemed to be the best strategy among the ones tested

Table 6.7: Characteristics of the instances tested for SMCFP.

Class Name	Nodes	Arcs	Commodities	Scenarios	Instances
p1_50	201	517	10	50	10
p2_50	201	501	10	50	10
p3_50	201	508	10	50	10
p4_50	201	520	10	50	10

Table 6.8: Summary of initial computational runs for SMCFP (averages over all 40 instances).

Algorithm Class	$\mathcal{L}$	# of iterations		# of		CPU time
		Major	Minor	Columns	Cuts	(sec.)
	1	2.8	441.0	8860.0	1582.8	1022.2
	5	2.8	480.9	8693.0	1086.0	336.7
AllCutsAllCols	10	2.5	520.8	9038.1	808.5	384.8
	25	2.8	749.9	9383.2	592.7	469.0
	50	5.0	1243.0	9081.3	527.6	829.8
	1	40.1	666.2	8706.6	2378.0	1158.0
	5	50.8	753.7	9295.8	1268.8	583.2
AllCutsOneCol	10	44.6	772.0	9208.5	942.0	471.5
	25	52.2	1012.8	9383.1	686.3	932.8
	50	42.0	1545.1	8881.4	617.6	1379.9
	1	47.3	502.2	10096.3	2478.4	1653.8
	5	61.7	553.7	10229.5	932.6	563.0
OneCutAllCols	10	67.1	152.6	9941.5	714.4	609.9
	25	130.1	1103.1	10908.0	451.7	891.4
	50	262.8	1899.8	11511.0	333.3	1477.8

Table 6.9: Summary of tests on switching criteria and cut-aggregation strategies for SMCFP (averages over all 40 instances).

Algorithm Class	Parameters			# of iterations		# of		CPU time
	$\mathcal{L}$	$\mathcal{P}$	$\mathcal{P}_{factor}$	Major	Minor	Columns	Cuts	(sec.)
AllCutsAllCols	5	-	-	2.8	480.9	8693.0	1086.0	336.7
FirstP	5	.50	.25	5.0	588.1	8814.1	934.6	187.3
BestP	5	.50	.25	4.8	537.2	9022.2	940.7	288.4
LastP	5	.50	.25	4.1	564.1	8960.2	960.6	220.4

initially. We decided to drop column generation on the extensive form, *AllCutsOneCol* and *OneCutAllCols* from further tests, and to set $\mathcal{L} = 5$ for the remainder of the tests.

Next, we focused on threshold switching strategies and how they compared to *AllCutsAllCols*. We used the settings that worked for our tests in Section 6.4.1.1, and hence, set $\mathcal{P} = .50$ and $\mathcal{P}_{factor} = .25$. Table 6.9 compares the performances of our threshold strategies using these cut-aggregation settings. Table 6.9 shows that algorithms using threshold switching criteria and cut-aggregation dominate the *AllCutsAllCols* approach. After examining these results, *AllCutsAllCols* was excluded from further testing.

For tests on stabilization, we used the parameter values that we found to be useful in Section 6.4.1.1. We used identical initialization parameters for both first- and second-stage column generation stabilization routines. The results of these tests are presented in Table 6.10. The table is sorted in ascending order of average solution times, and also displays information on the number of times an algorithm was the fastest to solve a problem instance and the number of times it was in the top 3.

Table 6.10: Summary of tests on stabilization strategies for SMCFP (with settings $\mathcal{L} = 5, \mathcal{P} = .50, \mathcal{P}_{factor} = .25$).

Algorithm		Iterations		# of		CPU time	# of times	
Switching	Stabilization	Maj	Min	Col	Cut	(sec.)	Top 3	Fastest
FirstP	*StabilizedCol&CutGen*[a]	5.2	787.5	8554.2	976.8	160.8	22	8
FirstP	*StabilizedCutGen*[b]	5.0	873.1	10538.4	881.9	162.1	10	1
LastP	*StabilizedColGen*[c]	3.4	756.8	8313.2	1151.4	172.7	12	2
FirstP	*StabilizedColGen*	4.3	781.1	8334.9	1052.0	175.2	16	12
FirstP	-	5.0	588.1	8814.1	934.6	187.3	14	2
LastP	*StabilizedCutGen*	4.1	803.0	9553.2	998.1	193.0	10	3
LastP	*StabilizedCol&CutGen*	3.9	818.8	8915.8	1101.2	197.5	12	3
BestP	*StabilizedColGen*	4.4	740.5	8685.5	1142.7	205.2	10	3
LastP	-	4.1	564.1	8960.2	960.6	220.4	7	4
BestP	*StabilizedCutGen*	4.8	797.2	10058.8	934.8	248.0	4	2
BestP	*StabilizedCol&CutGen*	4.8	827.5	10304.3	995.5	261.6	2	0
BestP	-	4.8	537.2	9022.2	940.7	288.4	1	0

[a]Stabilized column and cut generation with $\varepsilon_0 = 0.001, \varepsilon_{RF} = 1000, \varepsilon_{IF} = 1.1; \Delta_0 = 10, \Delta_{High} = 100, \epsilon_{acceptZ} = 0.01$.

[b]Stabilized cut generation with $\Delta_0 = 100, \Delta_{High} = 1000, \epsilon_{acceptZ} = 0.001$.

[c]Stabilized column generation with $\varepsilon_0 = 0.001, \varepsilon_{RF} = 100, \varepsilon_{IF} = 1.1$.

6.5 CONCLUSIONS

In this chapter we developed a new method that flexibly combines column generation with the L-shaped method for two-stage stochastic linear programs. This new method gives rise to a wide variety of algorithmic strategies that are guaranteed to converge under mild conditions, and brings computational advantages for large-scale problems. Particularly for those models where column generation constitutes a natural solution approach, this method brings a novel approach for computationally effective solutions.

We have tested the performances of various different algorithmic strategies on a number of instances of a two-stage stochastic cutting stock problem and a two-stage stochastic multi-commodity flow problem. These computational experiments suggest that the threshold algorithms, especially *LastP* and *FirstP*, combined with L-shaped cut aggregation are fast and effective. Since there are virtually infinitely many strategies that can be considered, it is not possible to reach concrete conclusions on what type of strategy may be the best.

But our findings confirm that strategies like threshold switching, L-shaped cut aggregation, multiple pricing and stabilization are of great help in designing computationally effective algorithms.

The method described in this chapter allows column generation within a two-stage stochastic linear programming setting. However, an extension to a branch-cut-and-price framework is possible, and exploring the performance of our algorithmic approaches in a stochastic integer programming setting is left for future research.

7.0 CONCLUSIONS AND FUTURE RESEARCH

This dissertation focuses on the design of regions for the liver allocation hierarchy. It introduces multi-objective integer programming and two-stage stochastic mixed-integer programming models to analyze various aspects of the system. The primary goal is to maximize organ utilization and transplant benefits, and minimize organ wastage through redesigning the regional configuration of OPOs across the U.S. under current liver allocation policies.

We formulate set-partitioning based mathematical models using binary variables that represent different region designs with the objective of finding the optimal partitioning of OPOs into a distinct set of regions with respect to different objectives. Column generation is the common methodology utilized in all our region design models, and we generate promising regions as needed throughout the solution procedure. We modify the ESLD and liver allocation system simulation model of Shechter et al. [155] for parameter estimation, scenario generation and solution validation purposes.

In Chapter 3, we introduce two parametric integer programming models based on the ε-constraint method for balancing the two objectives of maximizing efficiency and maximizing equity while designing regions for the liver allocation system. We extend previous work [96, 157] in a number of ways by utilizing a refined equity estimate, providing exact methods to obtain the Pareto frontier and embedding branch and price as a solution method within the multi-objective integer programming framework. Simulation analysis for our proposed regional configurations shows an average 5% increase in efficiency and 70% increase in equity levels when compared to current system performance, and also shows that our results dominate configurations provided by Stahl [157] and Kong et al. [97]. We also have a working paper [50] based on this chapter.

In Chapter 4, we introduce a stochastic programming model in order to incorporate uncertainty while modeling the national waiting list. We use a column generation approach and formulate a pricing problem that is capable of ranking patients on the waiting list under different scenarios while generating promising regions for the master problem. We use sampling methods to generate scenarios. In addition to relaxing the steady-state assumption which is common for all previous liver transplant region design studies [96, 97, 157], we also adopt a different objective: maximizing the expected life-time gained per transplant.

Chapter 5 introduces a modification of the stochastic mixed-integer program of Chapter 4 by switching to an OPO-based model, and replaces individual patients in every OPO with aggregate patients for every scenario. This aggregate model enables us to solve the problem using larger samples of scenarios and more batches, which results in tighter estimated optimality gaps for candidate solutions found using the sample average approximation method. According to our simulation analysis, the proposed region configurations of Chapters 4 and 5 bring up to a 7% increase in expected transplant outcomes for patients. Demirci et al. [49] is based on Chapters 4 and 5.

Motivated by Chapters 4 and 5, Chapter 6 introduces a decomposition framework that embeds column generation within the L-shaped method for two-stage stochastic linear programs with a large number of variables. In Chapters 4 and 5, we solve two-stage stochastic region-design problems using column generation. The fact that we model the ranking policies of UNOS within regions makes second-stage decisions automatic and enables a closed-form expression of the second-stage objectives. Consequently, this eliminates the need for a mechanism combining column and cut generation to be used. If our models allowed for decisions on how to rank patients, there would be a need to generate cuts and columns simultaneously for the master problem. Based on this observation, Chapter 6 deals with the general question of how to design effective implementations of column generation methods within a two-stage stochastic programming framework. We present many different algorithmic strategies, including stabilization techniques for both column and cut generation, and utilize two-stage stochastic versions of the cutting-stock and multi-commodity flow problems for extensive computational experiments. Demirci et al. [51] is based on this chapter.

7.1 FUTURE RESEARCH

In this section, we present several potential future research directions related to the models and solution techniques discussed in this dissertation.

Alternative models for equity: In Chapter 3, we model the equity level of the liver allocation system using a mini-max setting and focus on *geographic equity*, as measured by the intra-OPO transplant rate per patient in every OPO. However, there are many other definitions of equity, such as racial/ethnic or socieconomic equity. For instance, policy makers may want to make sure that system performance is similar across different ethnic groups. For future research, we plan to formulate models that incorporate alternative equity measures for the liver allocation system. In addition to this, this study focuses on an equity measure at the OPO level. In the future, we plan to design region-based equity measures that look at the overall equity levels within organ allocation regions.

Incorporating equity within a stochastic programming framework: In this dissertation, we study equity under a deterministic multi-objective programming setting. Furthermore, the stochastic programming models of Chapters 4 and 5 only focus on the efficiency of the model in terms of transplant outcomes per recipient. Our future research plans include formulating models that incorporate equity within a stochastic programming framework. A later extension might also look at extending these models into a multi-objective stochastic programming setting, in which both efficiency and equity could be analyzed under uncertainty.

Extending the region design models for other types of organs: In this dissertation, we study the liver allocation system and introduce models for designing liver transplant regions. The models discussed in this study can also be applied to other types of organs. In fact, an interesting application would be looking at how optimal regional configurations would change for different types of transplantable organs.

Extending the multi-objective optimization model: We propose a multi-objective programming framework to balance two objectives in the liver allocation system: maximizing the viability-adjusted number of transplants and maximizing the minimum geographic equity measure across all OPOs. A future research idea is to expand this multi-objective framework to incorporate the objective adopted in Chapters 4 and 5, i.e., maximizing the expected survival outcome per transplant.

Designing new modeling and solution techniques for the patient-based stochastic programming model of Chapter 4: The regional configuration that performed the best during simulation analysis for Chapters 4 and 5 was 4J, i.e., a solution found using the patient-based model. Since this model adopts a more realistic representation of liver offers, it might be expected to generate better results than those of Chapter 5 when solved over a sufficiently large sample of scenarios. However, as we discussed previously, the patient-based model is very hard to solve. Hence, our future research will focus on designing new modeling and solution approaches for this model, including polyhedral studies, generating valid inequalities, an exploring decomposition approaches, in order to test the performance of the model on larger scenario samples.

Using full branch and price for the stochastic programming models introduced in Chapters 4 and 5: We solve the models introduced in Chapters 4 and 5 using the SPRINT approach, and hence, use column generation only in the root node of the branch-and-bound tree while solving the models. In the future we will use full branch-and-price approaches to solve these models, i.e., with column generation in every node of the branch-and-bound tree. It would be interesting to compare the results obtained with SPRINT to those found with full branch-and-price, and analyze the trade off between solution quality and CPU time.

Extending the combined cut and column generation framework of Chapter 6 to stochastic integer programming: In this dissertation, we design a general framework that combines column generation and the L-shaped methods for two-stage stochastic linear programs. In the future, we will extend this method to a two-stage stochastic integer programming setting by embedding the procedure within branch and price.

Extending the combined cut and column generation framework of Chapter 6 to a multi-stage setting: Our research plans also include extending our combined cut and column generation framework to multi-stage stochastic linear programming. We will construct algorithms that combine column generation and nested Benders decomposition [21, 132] for multi-stage stochastic linear programs, and explore computational strategies, such as stabilization, within this framework.

BIBLIOGRAPHY

[1] B. Adenso-Diaz and F. Rodriguez. A simple search heuristic for the MCLP: Application to the location of ambulance bases in a rural region. *Omega-International Journal Of Management Science*, 25(2):181–187, 1997.

[2] J.-H. Ahn and J. C. Hornberger. Involving patients in the cadaveric kidney transplant allocation process: A decision-theoretic perspective. *Management Science*, 42(5):629–641, 1996.

[3] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, Englewood Cliffs, NJ, 1993.

[4] O. Alagoz, L. M. Maillart, A. J. Schaefer, and M. S. Roberts. The optimal timing of living-donor liver transplantation. *Management Science*, 50(10):1420–1430, 2004.

[5] O. Alagoz, L. M. Maillart, A. J. Schaefer, and M. S. Roberts. Choosing among living-donor and cadaveric livers. *Management Science*, 53(11), 2007.

[6] O. Alagoz, L. M. Maillart, A. J. Schaefer, and M. S. Roberts. Determining the acceptance of cadaveric livers using an implicit model of the waiting list. *Operations Research*, 55(1):24–36, 2007.

[7] O. Alagoz, M. S. Roberts, C. L. Bryce, A. J. Schaefer, J. Chang, and D. C. Angus. Incorporating biological natural history in simulation models: Empiric estimates of the progression of end-stage liver disease. *Medical Decision Making*, 25(6):620–632, 2005.

[8] O. Alagoz, A. J. Schaefer, and M. S. Roberts. Optimization in organ allocation. In P. Pardalos and E. Romeijn, editors, To appear in *Handbok of Optimization in Medicine*. Kluwer Academic Publishers, 2005.

[9] R. Anbil, R. Tanga, and E. L. Johnson. A global approach to crew-pairing optimization. *IBM Systems Journal*, 31(1):71–78, 1992.

[10] J. L. Arthur and A. Ravidran. A multiple objective nurse scheduling model. *IIE Transactions*, 13(1):55–60, 1981.

[11] J. R. Baker, E. R. Clayton, and L. J. Moore. Redesign of primary response areas for county ambulance services. *European Journal of Operational Research*, 41(1):23–32, 1989.

[12] C. Barnhart, A. Cohn, E. L. Johnson, D. Klabjan, G. L. Nemhauser, and P. H. Vance. Airline crew scheduling. In R.W. Hall, editor, *Handbook of Transportation Science*. Kluwer Academic Publishers, Norwell, MA, 2nd edition, 2002.

[13] C. Barnhart, C. A. Hane, and P. H. Vance. Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Operations Research*, 48(2):318–326, 2000.

[14] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3):316–329, 1998.

[15] E. M. L. Beale. On minimizing a convex function subject to linear inequalities. *Journal of the Royal Statistical Society Series B*, 17(2):173–184, 1955.

[16] H. Ben Amor and J. M. Valério de Carvalho. Cutting stock problems. In G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors, *Column Generation*. Springer, New York, 2005.

[17] J. F. Benders. Partitioning procedures for solving mixed variables programming problems. *Numerische Mathematics*, 4(1):238–252, 1962.

[18] J. M. Benedict. *Three Hierarchical Objective Models Which Incorporate the Concept of Excess Coverage to Locate EMS Vehicles or Hospitals*. PhD thesis, Northwestern University, Evanston, IL, 1983.

[19] P. Beraldi, M. E. Bruni, and D. Conforti. Designing robust emergency medical service via stochastic programming. *European Journal Of Operational Research*, 158(1):183–193, 2004.

[20] I. Berrada, J. A. Ferland, and P. Michelon. A multi-objective approach to nurse scheduling with both hard and soft constraints. *Socio-Economic Planning Science*, 30(3):183–193, 1996.

[21] J. R. Birge. Decomposition and partitioning methods for multistage stochastic linear programs. *Operations Research*, 33(5):989–1007, 1985.

[22] J. R. Birge and F. V. Louveaux. A multicut algorithm for two stage stochastic linear programs. *European Journal of Operational Research*, 34(3):384–392, 1988.

[23] J. R. Birge and F. V. Louveaux. *Introduction to Stochastic Programming*. Springer, New York, 1997.

[24] R. E. Bixby, J. W. Gregory, I. J. Lustig, R. E. Marsten, and D. F. Shanno. Very-large scale linear programming: A case study in combining interior point and simplex methods. *Operations Research*, 40(5):885–897, 1992.

[25] P. W. G. Bots and J. A. M. Hulshof. Designing multi-criteria decision analysis processes for priority setting in health policy. *Journal of Multi-Criteria Decision Analysis*, 9(1-3):56–75, 2000.

[26] C. C. Branas, E. J. MacKenzie, and C. S. ReVelle. A trauma resource allocation model for ambulances and hospitals. *Health Services Research*, 35(2):489–507, 2000.

[27] M. L. Brandeau, F. Sainfort, and W. P. Pierskalla. *Operations Research and Health Care: A Handbook of Methods and Applications*. Kluwer Academic Publishers, Boston, MA, 2004.

[28] L. Brotcorne, G. Laporte, and F. Semet. Ambulance location and relocation models. *European Journal of Operations Research*, 147(3):451–463, 2003.

[29] L. Brunetta, M. Conforti, and M. Fischetti. Polyhedral approach to an integer multicommodity flow problem. *Discrete Applied Mathematics*, 101(1):13–36, 2000.

[30] Centers for Medicare and Medicaid Services (CMS). National Health Expenditure Data: NHE Fact Sheet. Available from http://www.cms.hhs.gov/nationalhealthexpenddata/25_nhe_fact_sheet.asp, information and data accessed on April 13, 2008.

[31] Centers for Medicare and Medicaid Services (CMS). National Health Expenditure Projections 2007-2017. Available from http://www.cms.hhs.gov/nationalhealthexpenddata/downloads/proj2007.pdf, information and data accessed on April 13, 2008.

[32] V. Chankong and Y. Y. Haimes. *Multiobjective Decision Making Theory and Methodology*. Elsevier Science, New York, NY, 1983.

[33] A. Charnes and W. W. Cooper. *Management Models and Industrial Applications of Linear Programming*. Wiley, New York, NY, 1961.

[34] A. Charnes, W. W. Cooper, and R. Ferguson. Optimal estimation of executive compensation by linear programming. *Management Science*, 1(2):138–151, 1955.

[35] J. B. Christainson. Balancing policy objectives in long-term care. *Health Services Research*, 13(2):157–1 70, 1983.

[36] R. Church, P. Sorensen, and W. Corrigan. Manpower deployment in emergency services. *Fire Technology*, 55(2):420–427, 2003.

[37] COIN-OR. http://www.coin-or.org.

[38] COIN-OR. COIN-OR Documentation: Branch-Cut-Price Framework. Available from http://www.coin-or.org/documentation.html.

[39] CORE - Center for Organ Recovery and Education. http://www.core.org, information and data accessed on November 7, 2007.

[40] CONSAD Research Corporation. An analysis of alternative national policies for allocating donor livers for transplantation. Technical report, CONSAD Research Corporation, Pittsburgh, PA, 1995.

[41] G. B. Dantzig. Linear programming under uncertainty. *Management Science*, 1(3-4):197–206, 1955.

[42] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8(1):101–111, 1960.

[43] M. S. Daskin. *Network and Discrete Location: Models, Algorithms and Applications*. Wiley, New York, NY, 1995.

[44] M. S. Daskin and L. K. Dean. Locations of health care facilities. In M. L. Brandeau, F. Sainfort, and W. P. Pierskalla, editors, *Operations Research and Health Care: A Handbook of Methods and Applications*. Kluwer Academic Publishers, Boston, MA, 2004.

[45] I. David and U. Yechiali. A time-dependent stopping problem with application to live organ transplants. *Operations Research*, 33(3):491–504, 1985.

[46] I. David and U. Yechiali. Sequential assignment match processes with arrivals of candidates and offers. *Probability in the Engineering and Informational Sciences*, 4(4):413–430, 1990.

[47] I. David and U. Yechiali. One-attribute sequential assignment match processes in discrete time. *Operations Research*, 43(5):879–884, 1995.

[48] I. Deák. Multidimensional integration and stochastic programming. In Y. Ermoliev and R. Wets, editors, *Numerical Techniques for Stochastic Optimization*. Springer-Verlag, Berlin, 1988.

[49] M. C. Demirci, A. J. Schaefer, and M. S. Roberts. Designing liver transplant regions under uncertainty. Technical report, University of Pittsburgh, Department of Industrial Engineering, 2008.

[50] M. C. Demirci, A. J. Schaefer, H. E. Romeijn, and M. S. Roberts. Balancing efficiency and equity in the liver allocation hierarchy. Technical report, University of Pittsburgh, Department of Industrial Engineering, 2008.

[51] M. C. Demirci, A. J. Schaefer, and J. M. Rosenberger. Column generation within the L-shaped method for stochastic linear programs. Technical report, University of Pittsburgh, Department of Industrial Engineering, 2008.

[52] C. Derman, G. J. Lieberman, and S. M. Ross. A sequential stochastic assignment problem. *Management Science*, 18(7):349–355, 1972.

[53] J. Desrosiers, F. Soumis, and M. Desrochers. Routing with time windows by column generation. *Networks*, 14(4):545–565, 1984.

[54] K. Doerner, A. Focke, and W. J. Gutjahr. Multicriteria tour planning for mobile healthcare facilities in a developing country. *European Journal of Operational Research*, 179(3):1078–1096, 2007.

[55] O. du Merle, D. Villeneuve, J. Desrosiers, and P. Hansen. Stabilized column generation. *Discrete Mathematics*, 194(1-3):229–237, 1999.

[56] D. J. Eaton, M. S. Daskin, D. Simmons, B. Bulloch, and G. Jansma. Determining emergency medical service vehicle deployment in Austin, Texas. *Interfaces*, 15(1):96–108, 1985.

[57] D. J. Eaton, H. M. Sanchez, R. R. Lantigua, and J. Morgan. Determining ambulance deployment in Santo-Domingo, Dominican-Republic. *Journal Of The Operational Research Society*, 37(2):113–126, 1986.

[58] M. Ehrgott. *Multicriteria optimization*. Springer, Berlin, 2005.

[59] M. Ehrgott. A discussion of scalarization techniques for multiple objective integer programming. *Annals of Operations Research*, 147(1):343–360, 2006.

[60] M. Ehrgott and X. Gandibleux. A survey and annotated bibliography of multiobjective combinatorial optimization. *OR Spektrum*, 22(4):425–460, 2000.

[61] M. Ehrgott and D. Ryan. Constructing robust crew schedules with bicriteria optimization. *Journal of Multi-Criteria Decision Analysis*, 11(3):139–150, 2003.

[62] M. Ehrgott and J. Tind. Column generation in integer programming with applications in multicriteria optimization. Technical report, The University of Auckland, Department of Engineering Science, 2007.

[63] M. Ehrgott and M. M. Wiecek. Multiobjective programming. In J. Figueria, S. Greco, and M. Ehrgott, editors, *Multiple Criteria Decision Analysis: State of the Art Surveys*. Kluwer Academic Publishers, Boston, MA, 2005.

[64] L. R. Ford and D. R. Fulkerson. A suggested computation for maximal multicommodity network flows. *Management Science*, 5(1):97–101, 1958.

[65] L. S. Franz, T. R. Rakes, and A. S. Wynne. A chance-constrained multiobjective model for mental health services planning. *Socio-Economic Planning Science*, 18(2):89–95, 1984.

[66] V. Gascon, S. Villeneuve, P. Michelon, and J. A. Ferland. Scheduling the flying squad nurses of a hospital using a multi-objective programming model. *Annals of Operational Research*, 96:149–166, 2000.

[67] H. I. Gassmann. MSLiP: A computer code for the multistage stochastic lienar programming problem. *Mathematical Programming*, 47:407–423, 1990.

[68] M. Gendreau, G. Laporte, and F. Semet. The maximal expected coverage relocation problem for emergency vehicles. *Journal Of The Operational Research Society*, 57(1):22–28, 2006.

[69] P. Ghandfaroush. Optimal allocation of time in a hospital pharmacy using goal programming. *European Journal of Operational Research*, 70:191–198, 1993.

[70] P. C. Gilmore and R. E. Gomory. A linear programming approach to the cutting stock problem. *Operations Research*, 9:849–859, 1961.

[71] G. D. Glockner and G. L. Nemhauser. A dynamic network flow problem with uncertain arc capacities: fomulation and problem structure. *Operations Research*, 48(2):233–242, 2000.

[72] Government Accounting Office. Current Policies and Practices. Available from http://www.gao.gov/special.pubs/organ/chapter2.pdf, information and data accessed on April 13, 2008, 2003.

[73] P. Griffin, M. Savelsbergh, and J. L. Swann. The health care system: Operations research and improving access. Technical report, Georgia Institute of Technology, School of Industrial and Systems Engineering, 2007.

[74] J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms II*. Comprehensive Studies in Mathematics. Springer-Verlag, 1993.

[75] A. Holder. Radiotherapy treatment design and linear programming. In M. L. Brandeau, F. Sainfort, and W. P. Pierskalla, editors, *Operations Research and Health Care: A Handbook of Methods and Applications*. Kluwer Academic Publishers, Boston, MA, 2004.

[76] K. Holmberg. Generalized cross decomposition with variable and constraint duplication techniques. Technical Report LiTH-MAT-R-1988-17, Linköping Institute of Technology, Department of Mathematics, 1988.

[77] K. Holmberg. Cross decomposition and Lagrangean relaxation of the Benders master probles. Technical Report LiTH-MAT-R-1989-14, Linköping Institute of Technology, Department of Mathematics, 1989.

[78] K. Holmberg. On the convergence of cross decomposition. *Mathematical Programming*, 47(1-3):269–296, 1990.

[79] K. Holmberg. Cross decomposition applied to integer programming problems: Duality gaps and convexification in parts. *Operations Research*, 42(4):657–668, 1994.

[80] J. C. Hornberger and J.-H. Ahn. Deciding eligibility for transplantation when a donor kidney becomes available. *Medical Decision Making*, 17(2):160–170, 1997.

[81] D. H. Howard. Why do transplant surgeons turn down organs? A model of the accept/reject decision. *Journal of Health Economics*, 21(2):957–969, 2002.

[82] J. Hu and E. L. Johnson. Computational results with a primal-dual subproblem simplex method. *Operations Research Letters*, 25(4):149–157, 1999.

[83] ILOG. *CPLEX 9.0 user's manual*. 2003.

[84] ILOG CPLEX: High Performance Software for Mathematical Programming and Optimization. http://www.ilog.com/products/cplex/.

[85] D. A. Jacobs, M. N. Silan, and B. A. Clemson. An analysis of alternative locations and service areas of American Red Cross blood facilities. *Interfaces*, 26(3):40–50, 1996.

[86] S. H. Jacobson and E. C. Sewell. Designing pediatric formularies for childhood immunization using inter programming models. In M. L. Brandeau, F. Sainfort, , and W. P. Pierskalla, editors, *Operations Research and Health Care: A Handbook of Methods and Applications*. Kluwer Academic Publishers, Boston, MA, 2004.

[87] S. H. Jacobson, E. C. Sewell, R. Deuson, and B. G. Weniger. An integer programming model for vaccine procurement and delivery for childhood immunization: a pilot study. *Health Care Management Science*, 2(1):1–9, 1999.

[88] P.S. Kamath, R.H. Wiesner, M. Malinchoc, W. Kremers, T.M. Therneau, C.L. Kosberg, G. D'Amico, E.R. Dickson, and W.R. Kim. A model to predict survival in patients with end-stage liver disease. *Hepatology*, 33(2):464–70, 2001.

[89] L.V. Kantorovich. Mathematical methods of organizing and planning production (translation of a report presented to Leningrad State University May 1939). *Management Science*, 6(4):366–422, 1960.

[90] E. P. C. Kao and M. Queyranne. Budgeting costs of nursing in a hospital. *Management Science*, 31(5):608–621, 1985.

[91] K. E. Kendall and S. M. Lee. Formulating blood rotation policies with multiple objectives. *Management Science*, 26(11):1145–1157, 1980.

[92] A. J. Keown and J. D. Martin. An integer goal programming model for capital budgeting in hospitals. *Financial Management*, 5(3):28–35, 1976.

[93] D. Kim and C. Barnhart. Multimodal express shipment service design: Models and algorithms. *Computers and Industrial Engineering*, 33(3):685–688, 1997.

[94] K. Kiwiel. Proximity control in bundle methods for convex nondifferentiable minimization. *Mathematical Programming*, 46(1-3):105–122, 1990.

[95] A. J. Kleywegt, A. Shapiro, and T. Homem De Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502, 2001.

[96] N. Kong. *Optimizing the Efficiency of the United States Organ Allocation System Through Region Reorganization*. PhD thesis, University of Pittsburgh, Pittsburgh, PA, 2006.

[97] N. Kong, A.J. Schaefer, B. Hunsaker, and M. S. Roberts. Maximizing the efficiency of the U.S. liver allocation system through region design. Technical report, Purdue University, Weldon School of Biomedical Engineering, 2007.

[98] E. V. Krichagina, R. Rubio, M. I. Taksar, and L. Wein. A dynamic stochastic stock-cutting problem. *Operations Research*, 46(5):690–701, 1998.

[99] M. Lamiri, X. Xie, A. Dolgui, and F. Grimaud. A stochastic model for operating room planning with elective and emergency demand for surgery. *European Journal Of Operational Research*, 185:1026–1037, 2008.

[100] M. Langer, R. Brown, M. Urie, J. Leong, M. Stracher, and J. Shapiro. Large scale optimization of beam weights under dose-volume restrictions. *International Journal of Radiation Oncology, Biology, Physics*, 18(4):887–893, 1990.

[101] M. Langer and J. Leong. Optimization of beam weights under dose-volume restrictions. *International Journal of Radiation Oncology, Biology, Physics*, 13(8):1255–1260, 1987.

[102] C. W. Lee and N. K. Kwak. Information resource planning for a health-care system using an AHP-based goal programming method. *Journal of the Operational Research Society*, 50(12):1191–1198, 1998.

[103] E. K. Lee, T. Fox, and I. Crocker. Optimization of radiosurgery treatment planning via mixed-integer programming. *Medical Physics*, 27(5):995–1004, 2000.

[104] E. K. Lee, R. J. Gallagher, D. Silvern, C. S. Wuu, and M. Zaider. Treatment planning for brachytherapy. An integer programming model, two computational approaches and experiments with permanent implant planning. *Physics in Medicine and Biology*, 44(1):145–165, 1999.

[105] E. K. Lee and M. Zaider. Determining an effective planning volume for permanent prostate implants. *International Journal of Radiation Oncology, Biology, Physics*, 49(4):1197–1206, 2001.

[106] E. K. Lee and M. Zaider. Intra-operative dynamic dose optimization in permanent prostate implants. *International Journal of Radiation Oncology, Biology, Physics*, 56(3):854–861, 2003.

[107] E. K. Lee and M. Zaider. Mixed integer programming approaches to treatment planning for brachytherapy - application to permanent prostate implants. *Annals of Operations Research*, 119(1-4):147–163, 2003.

[108] J. Linderoth and S. Wright. Decomposition algorithms for stochastic programming on a computational grid. *Computational Optimization and Applications*, 24(2-3):207–250, 2003.

[109] B. G. Lopez-Valcarel and P. B. Perez. Evaluation of alternative functional designs in an emergency department by means of simulation. *Simulation*, 63(1):20–28, 1994.

[110] M. E. Lübbecke and J. Desrosiers. Selected topics in column generation. *Operations Research*, 53(6):1007–1023, 2005.

[111] G. Lulli and S. Sen. A branch-and-price algorithm for multistage stochastic integer programming with application to stochastic batch-sizing problems. *Management Science*, 50(6):786–796, 2004.

[112] G. Lyberatos and E. M. Abulesz. Analysis of periodic injections in cancer chemotherapy. *International Journal of Systems Science*, 21(8):1659–1671, 1990.

[113] M. Malinchoc, P. S. Kamath, F. D. Gordon, C. J Peine, J. Rank, and P. L. ter Borg. A model to predict poor survival in patients undergoing transjugular intrahepatic portosystemic shunts. *Hepatology*, 31(4):864–871, 2000.

[114] A. Martel and J. Ouellet. Stochastic allocation of a resource among partially interchangeable activities. *European Journal Of Operational Research*, 74(3):528–539, 1994.

[115] R. K. Martin. *Large Scale Linear and Integer Optimization: A Unified Approach*. Kluwer Academic Press, 1999.

[116] W. E. McAleer and I. A. Naqvi. The relocation of ambulance stations - A successful case-study. *European Journal Of Operational Research*, 75(3):582–588, 1994.

[117] A. Mehrez, Z. Sinuany-Stern, A.-G. Tal, and B. Shemuel. On the implementation of quantitative facility location models: The case of a hospital in a rural region. *Journal of the Operational Research Society*, 47(5):612–625, 1996.

[118] A. Mehrotra, E. L. Johnson, and G. L. Nemhauser. An optimization based heuristic for political districting. *Management Science*, 44(8):1100–1114, 1998.

[119] R. M. Merion, M. K. Guidinger, J. M. Newmann, M. D. Ellison, F. K. Port, and R. A. Wolfe. Prevalence and outcomes of multiple-listing for cadaveric kidney and liver transplantation. *American Journal of Transplantation*, 4(1):94–100, 2004.

[120] H. E. Miller, W. P. Pierskalla, and G. J. Rath. Nurse scheduling using mathematical programming. *Operations Research*, 24(5):857–870, 1976.

[121] H. Minkowski. *Geometrie der Zahlen.* Teubner Leipzig, 1896.

[122] D. P. Morton and R. K. Wood. On a stochastic knapsack problem and generalizations. In D. L. Woodruff, editor, *Advances in Computational and Stochastic Optimization, Logic Programming, and Heuristic Search: Interfaces in Computer Science and Operations Research*, pages 149–168. Kluwer Academic Publishers, Dordrecht, the Netherlands, 1998.

[123] Multicommodity problems. Available from ftp://ftp.eng.auburn.edu/pub/pvance/data/mcf/mcf_data.

[124] A. A. Musa and U. Saxena. Scheduling nurses using goal-programming techniques. *IEE Transactions*, 16(3):216–221, 1984.

[125] S. Narashimhan, H. Pirkul, and D. A. Schilling. Capacitated emergency facility siting with multiple levels of backup. *Annals of Operations Research*, 40:323–337, 1992.

[126] National Center for Health Statistics (NCHS). National Vital Statistics Report - Deaths: Final Data for 2004. NVSR Vol. 55, No. 19. Available from http://www.cdc.gov/nchs/data/nvsr/nvsr55/nvsr55_19.pdf.

[127] C. A. Nelson and J. R. Wolch. Intrametroplitan planning for community based residential care: A goal programming approach. *Socio-Economic Planning Science*, 19(3):205–212, 1985.

[128] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization.* John Wiley and Sons, New York, NY, 1988.

[129] F. F. Nobre, L. T. F. Trotta, and L. F. A. M. Gomes. Multi-criteria decision making - An approach to setting priorities in heath care. *Statistics in Medicine*, 18,(23):3345–3354, 1999.

[130] I. Ozkarahan and J.E. Bailey. Goal programming model subsystem of a flexible nurse scheduling support system. *IEE Transactions*, 20(3):306–316, 1988.

[131] D. Parr and J.M. Thompson. Solving the multi-objective nurse scheduling problem with a weighted cost function. *Annals of Operations Research*, 155:279–288, 2007.

[132] M. V. F. Pereira and L. M. V. G. Pinto. Stochastic optimization of a multireservoir hydroelectric system - A decomposition approach. *Water Resources Research*, 21:779–792, 1985.

[133] W. P. Pierskalla and D. J. Brailer. Applications of operations research in health care delivery. In S. M. Pollock, M. H. Rothkopf, and A. Barnett, editors, *Handbooks on*

Operations Research and Management Sciences Vol. 6. Elsevier Science, Amsterdam, The Netherlands, 1994.

[134] H. Pirkul and D. A. Schilling. The siting of emergency service facilities with workload capacities and backup service. *Management Science*, 34(7):896–908, 1988.

[135] F. D. Preciado-Walters, R. L. Rardin, M. Langer, and V. Thai. A coupled column generation, mixed integer approach to optimal planning of intensity modulated radiation therapy for cancer. *Mathematical Programming*, 101(2):319–338, 2004.

[136] W. L. Price and M. Turcotte. Locating a blood-bank. *Interfaces*, 16(5):17–26, 1986.

[137] A. A. B. Pritsker. Life and death decisions: Organ transplantation allocation policy analysis. *OR/MS Today*, 25(4):22–28, 1998.

[138] P. Punnakitikashem, J. M. Rosenberger, and D. B. Behan. Stochastic programming for nurse assignment. To appear in *Computational Optimization and Applications*, 2006.

[139] M. L. Puterman. *Markov Decision Processes*. Wiley, New York, NY, 1994.

[140] T. K. Ralphs and L. Ladányi. COIN/OR user's manual. *Technical report, IBM T. J. Watson Reserach Center, Yorktown Heights, NY*, 2001.

[141] T. K. Ralphs, L. Ladányi, and M. J. Saltzman. Parallel branch, cut and price for large-scale discrete optimization. *Mathematical Programming*, 98(1-3):253–280, 2003.

[142] A. K. Rifai and J. O. Pecenka. An application of goal programming in healthcare planning. *International Journal of Production Management*, 10(3):28–37, 1989.

[143] R. Righter. A resource allocation problem in a random environment. *Operations Research*, 37(2):329–338, 1989.

[144] M.S. Roberts, D. C. Angus, C. L. Bryce, Z. Valenta, and L. Weissfeld. Survival after liver transplantation in the united states: a disease-specific analysis of the unos database. *Liver Transplantation*, 10(7):886–97, 2004.

[145] H. E. Romeijn, R. K. Ahuja, J. F. Dempsey, and A. Kumar. A column generation approach to radiation therapy treatment planning using aperture modulation. *SIAM Journal on Optimization*, 15(3):838–862, 2005.

[146] C. Romero and T. Rehman. A note on diet planning in the third world by linear and goal programming. *Journal of the Operational Research Society*, 35:555–558, 1984.

[147] A Ruszczyński. A regularized decomposition method for minimizing a sum of polyhedral functions. *Mathematical Programming*, 35(3):309–333, 1986.

[148] D. M. Ryan and A. B. Foster. An integer programming approach to scheduling. In A. Wren, editor, *Computer Scheduling of Public Transport Urban Passenger Vehicle and Crew Scheduling*. North-Holland, Amsterdam, The Netherlands, 1981.

[149] T. L. Saaty. *The Analytic Hierarchy Process*. McGraw Hill, New York, NY, 1980.

[150] B. Sandıkçı, L. M. Maillart, A. J. Schaefer, O. Alagoz, and M. S. Roberts. Estimating the patients price of privacy in liver transplantation. Technical report, University of Pittsburgh, Department of Industrial Engineering, 2008.

[151] C. Sapountzis. Allocating blood to hospitals. *Journal of the Operational Research Society*, 40(5):443–449, 1989.

[152] R.S. Segall. Some quantitative methods for determining capacities and locations of military emergency medical facilities. *Applied Mathematical Modelling*, 24(5-6):365–389, 2000.

[153] E. C. Sewell and S. H. Jacobson. Using an integer programming model to determine the price of combination vaccines for childhood immunization. *Annals Of Operations Research*, 119(1-4):261–284, 2003.

[154] A. Shapiro. Monte carlo sampling methods. In A. Ruszczynski and A. Shapiro, editors, *Stochastic Programming, Handbook in Operations Research and Management Science Vol. 10*. Elsevier Science, Amsterdam, 2003.

[155] S. M. Shechter, C. L. Bryce, O. Alagoz, J. E. Kreke, J. E. Stahl, A. J. Schaefer, D. C. Angus, and M. S. Roberts. A clinically based discrete-event simulation of end-stage liver disease and the organ allocation process. *Medical Decision Making*, 25(2):199–209, 2005.

[156] Z. Sinuany-Stern, A. Mehrez, A.-G. Tal, and B. Shemuel. The location of a hospital in a rural region: The case of the Negev. *Location Science*, 3(4):255–266, 1995.

[157] J. E. Stahl, N. Kong, S. M. Shechter, A. J. Schaefer, and M. S. Roberts. A methodological framework for optimally reorganizing liver transplant regions. *Medical Decision Making*, 25(1):35–46, 2005.

[158] C. Stummer, K. Doerner, A. Focke, and K. Heidenberger. Determining location and size of medical departments in a hospital network: A multiobjective decision support approach. *Health Care Management Science*, 7(1):63–71, 2004.

[159] X. Su and S. A. Zenios. Patient choice in kidney allocation: The role of the queueing discipline. *Manufacturing & Service Operations Management*, 6(4):280–301, 2004.

[160] X. Su and S. A. Zenios. Patient choice in kidney allocation: A sequential stochastic assignment model. *Operations Research*, 53(3):443–455, 2005.

[161] X. Su and S. A. Zenios. Recipient choice can address the efficiency-equity trade-off in kidney transplantation: A mechanism design model. *Management Science*, 52(1):1647–1660, 2006.

[162] M. Tamiz, D. Jones, and C. Romero. Goal programming for decision making: An overview of the current state-of-the-art. *European Journal of Operational Research*, 111(3):569–581, 1998.

[163] P. Toth and D. Vigo. *The Vehicle Routing Problem*. SIAM, 2002.

[164] E. Totsuka, J. J. Fung, M.C. Lee, T. Ishii, M. Umehara, Y. Makino, T. H. Chang, Y. Toyoki, S. Narumi, K. Hakamada, and M. Sasaki. Influence of cold ischemia time and graft transport distance on postoperative outcome in human liver transplantation. *Surgery Today*, 32(9):792–799, 2002.

[165] TransWeb. http://www.transweb.org, information and data accessed on November 7, 2007.

[166] V. M. Trivedi. A mixed-integer goal programming model for nursing service budgeting. *Operations Research*, 29(5):1019–1034, 1981.

[167] S. Trukhanov, L. Ntaimo, and A. J. Schaefer. On adaptive multicut aggregation for two-stage stochastic linear programs. Technical report, Texas A&M University, Department of Industrial and Systems Engineering, 2008.

[168] United Network for Organ Sharing (UNOS). Allocation of livers. Available from http://www.unos.org/policiesandbylaws2/policies/pdfs/policy_8.pdf, information and data accessed on April 13, 2008.

[169] United Network for Organ Sharing (UNOS). http://www.unos.org, information and data accessed on April 13, 2008.

[170] U.S. Department of Health and Human Services, Health Resouces and Services Administration. Partnering with Your Transplant Team: The Patient's Guide to Transplantation. Available from http://www.unos.org/sharedcontentdocuments/transplantation_guide_final-3-04-04.pdf.

[171] US Transplant - Scientific Registry of Transplant Recipients. http://www.ustransplant.org, information and data accessed on November 7, 2007.

[172] J. M. Valério de Carvalho and A. J. Guimarães Rodrigues. An LP-based approach to a two-stage cutting stock problem. *European Journal of Operations Research*, 84(3):580–589, 1996.

[173] T. J. Van Roy. Cross decomposition for mixed integer programming. *Mathematical Programming*, 25(1):46–63, 1983.

[174] R. Van Slyke and R. J-B. Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663, 1969.

[175] F. Vanderbeck. Computational study of a column generation algorithm for bin packing and cutting stock problems. *Mathematical Programming*, 86(3):565–594, 1999.

[176] F. Vanderbeck. Exact algorithm for minimizing the number of setups in the one-dimensional cutting stock problem. *Operations Research*, 48(6):915–926, 2000.

[177] F. Vanderbeck. Implementing mixed integer column generation. In Desaulniers G., Desrosiers J., and Solomon M. M., editors, *Column Generation*. Springer-Verlag, Boston,MA, 2005.

[178] D. M. Warner. Scheduling nursing personnel according to nursing preference: A mathematical programming approach. *Operations Research*, 24(5):842–856, 1976.

[179] R. H. Wiesner, S. V. McDiarmid, P. S. Kamath, E. B. Edwards, M. Malinchoc, W. K. Kremers, R. A. F. Krom, and W. R. Kim. MELD and PELD: Application of survival models to liver allocation. *Liver Transplantation*, 7(7):567–580, 2001.

[180] World Health Organization (WHO). Core Health Indicators. Available from http://www.who.int/whosis/database/core/core_select.cfm, information and data accessed on April 13, 2008.

[181] World Health Organization (WHO). World Health Report 2000. Available from http://www.who.int/whr/2000/en/whr00_en.pdf, information and data accessed on April 13, 2008.

[182] S. Yoo, M. E. Kowalok, B. R. Thomasden, and D. L. Henderson. Treatment planning for prostate brachytherapy using region of interest adjoint functions and a greedy heuristic. *Physics in Medicine and Biology*, 48(24):4077–4090, 2003.

[183] M. Zaider, M. Zelefsky, E. K. Lee, K. Zakian, H. A. Amols, J. Dyke, and J. Koutcher. Treatment planning for prostate implants using MR spectroscopy imaging. *International Journal of Radiation Oncology, Biology, Physics*, 47(4):1085–1096, 2000.

[184] S. A. Zenios. Modeling the transplant waiting list: A queuing model with reneging. *Queuing Systems*, 31(3-4):239–251, 1999.

[185] S. A. Zenios. Optimal control of a paired-kidney exchange program. *Management Science*, 48(3):328–342, 2002.

[186] S. A. Zenios, G.M. Chertow, and L.M. Wein. Dynamic allocation of kidneys to candidates on the transplant waiting list. *Operations Research*, 48(4):549–569, 2000.

[187] S. A. Zenios, L.M. Wein, and G.M. Chertow. Evidence-based organ allocation. *American Journal of Medicine*, 107(1):52–61, 1999.

[188] P. P. Zhang, J. Wu, D. Dean, L. Xing, J. Y. Xue, R. Maciunas, and C. Sibata. Plug pattern optimization for gamma knife radiosurgery treatment planning. *International Journal of Radiation Oncology Biology Physics*, 55(2):420–427, 2003.

[189] Z. Zhu and M. A. McKnew. A goal programming workload balancing optimization model for ambulance allocation: An application to Shanghai, P R. China. *Socio-Economic Planning Science*, 27(2):137–148, 1993.