ABSTRACT

ERDOGAN, SAADET AYCA. Optimization of Appointment Based Scheduling Systems. (Under the direction of Brian T. Denton.)

Appointment based service systems have been studied widely in the operations research and medical literature over the past three decades. However, the current literature ignores some important aspects of appointment scheduling. In this thesis, we begin by reviewing the existing literature on appointment scheduling with an emphasis on health care applications such as primary and specialty care clinical scheduling, and surgery scheduling. We point out areas in need of further research, particularly focusing on the uncertain nature of customer demand on service systems. We also provide a review of the stochastic programming literature which is relevant to the methodological aspect of this thesis.

We present new stochastic programming models of single server appointment scheduling systems which combine two important aspects of uncertainty: uncertainty in service durations and uncertainty in customer demand. The first model is a static scheduling model with a known number of customers each of which may no-show for their appointment. This problem is modeled as a two-stage stochastic linear program with allowances between customer appointments being the first stage decisions, and customer waiting times and server overtime being the second stage decisions. We provide theoretical properties that give insights into the structure of the optimal appointment schedules, and we present numerical results to demonstrate the effects of cost coefficients and no-show probabilities on the optimal schedule.

The second model considers a dynamic appointment scheduling problem in which an uncertain number of customers request appointments sequentially. The customers are assumed to be scheduled in a sequence that is fixed a priori. This problem is modeled as a multi-stage stochastic linear program with allowances between customer appointments being the first stage decisions, and customer waiting times and server overtime being the second stage decisions. We provide theoretical properties that give insights into the structure of the optimal appointment schedules, and we present numerical results to demonstrate the effects of cost coefficients and no-show probabilities on the optimal schedule.

The third model we present relaxes the assumption of a fixed sequence of customers. We
show with a counter-example that a first-come-first-serve based queuing discipline may not be optimal in some scheduling environments. In the presence of different customer types and priorities, the sequencing decisions must be incorporated within the appointment scheduling process. We present a two-stage stochastic integer program which captures the dynamic multistage nature of appointment requests as well as the sequencing of customers. We provide two alternative formulations and tested alternative implementations of the integer L-shaped method. We also present several approaches to tighten the formulation of the subproblems and improve computational efficiency of their solution. Our numerical experiments provide insights on the optimal sequencing and scheduling decisions as well as on the performance of the solution methods.
Optimization of Appointment Based Scheduling Systems

by
Saadet Ayca Erdogan

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Operations Research

Raleigh, North Carolina

2010

APPROVED BY:

Thom Hodgson

Yahya Fathi

Reha Uzsoy

Brian T. Denton
Chair of Advisory Committee
DEDICATION

To my father.
BIOGRAPHY

Saadet Ayca Erdogan was born in 1979, in Ankara, Turkey. She received her Bachelor of Science degree in Industrial Engineering from Istanbul Technical University, Turkey in 2001. Upon graduation she entered the Master of Science program in Industrial Engineering, also at Istanbul Technical University. She completed her degree requirements at 2004 and entered the graduate program in Operations Research at North Carolina State University, Raleigh, NC. She married Erdem Serkan Erdogan in 2004. They currently reside in Sunnyvale, California.
ACKNOWLEDGEMENTS

This thesis would not have been possible without the invaluable support and guidance of my advisor Dr. Brian Denton, who always believed in me and encouraged me to succeed during this challenging process. I am very thankful to him for his endless support. I would also like to acknowledge the support for this research which was provided in part by grant CMMI-0620573 from the National Science Foundation.

I would like to thank my committee members, Dr. Thom Hodgson, Dr. Reha Uzsoy and Yahya Fathi, without whose encouragements I would not be in this Operations Research program, for agreeing to be in my committee and providing feedback on my work.

I would also like to thank my friends Alex Gose, Bjorn Berg and Daniel Underwood for their collaboration, insightful comments and technical assistance in this work. Special thanks go to my dear friends Natalia, Sebnem, Jennifer and Emine for always being there when I needed and for cheering me up.

Last, but not least, I thank my family: my father, who started planting the idea of getting a PhD into my mind from the first day of college, my mother, my sisters and my little brother and my in-laws for their unconditional love and support. And my husband Serkan, for sharing his experiences in PhD, for listening to my continuous complaints and frustrations with patience, and for his endless love.
# TABLE OF CONTENTS

List of Tables .......................................................... vii
List of Figures ........................................................... ix

Chapter 1 Introduction .................................................. 1
  1.1 Research Objective .............................................. 2
  1.2 Summary of Research Contributions ............................ 3
  1.3 Organization of the Document ................................. 3

Chapter 2 Literature Review ......................................... 5
  2.1 Appointment Based Scheduling Systems ......................... 5
    2.1.1 Queuing Models .......................................... 7
    2.1.2 Simulation ............................................... 8
    2.1.3 Optimization ........................................... 10
    2.1.4 Heuristics .............................................. 12
  2.2 Literature Review on Stochastic Programming .................. 14
    2.2.1 Stochastic Linear Programs and Solution Methods .......... 15
    2.2.2 Multi-stage Stochastic Linear Programs .................... 17
    2.2.3 Methods to Accelerate Convergence ....................... 19
  2.3 Specific Contributions to the Literature ...................... 20

Chapter 3 Appointment Scheduling in the Presence of No-Shows ... 23
  3.1 Model Formulation ............................................. 25
  3.2 Structural Properties and Solution Methodology ............... 27
    3.2.1 No-Shows and Double-Booking .......................... 27
    3.2.2 Subproblem Structure .................................. 29
    3.2.3 L-Shaped Method ...................................... 30
  3.3 Results ........................................................ 31
  3.4 Conclusions .................................................. 33

Chapter 4 Dynamic Appointment Scheduling with Uncertain Demand ... 35
  4.1 Dynamic Appointment Scheduling ............................... 36
    4.1.1 Simple Examples ....................................... 37
    4.1.2 Model Formulation ..................................... 38
  4.2 Structural Properties and Solution Methodology ................. 41
    4.2.1 Subproblem Structure .................................. 41
    4.2.2 Valid Inequalities ..................................... 42
    4.2.3 Multi-Cut Outer Linearization: ........................ 45
    4.2.4 Nested Decomposition .................................. 46
  4.3 Results ........................................................ 47
    4.3.1 Structure of the Optimal solution ...................... 48
    4.3.2 Computational Performance of Proposed Methods .......... 50
### 4.3.3 Convergence of the Lower Bound .......................... 52
### 4.3.4 Value of the Stochastic Solution .......................... 54
### 4.4 Conclusions ................................................. 54

#### Chapter 5 Dynamic Appointment Sequencing and Scheduling ........ 56

* 5.1 Motivation for Dynamic Appointment Sequencing and Scheduling .... 59
  * 5.1.1 Dynamic Appointment Scheduling Model with AOFS ............... 60
  * 5.1.2 Counter-Example: FCFS vs. AOFS .................................. 64
* 5.2 Dynamic Appointment Sequencing and Scheduling Model .............. 65
* 5.3 Structural Properties and Solution Methodology ........................ 71
  * 5.3.1 Alternative Formulation ........................................... 72
  * 5.3.2 Solution to Scenario Subproblems .................................. 72
  * 5.3.3 Big M Values ....................................................... 73
  * 5.3.4 Valid Inequalities .................................................. 74
  * 5.3.5 Integer L-Shaped Method ........................................... 75
* 5.4 Results .................................................................. 77
  * 5.4.1 Structure of the Optimal Solution .................................... 77
  * 5.4.2 Computational Performance of Proposed Methods .................. 81
* 5.5 Conclusions ......................................................... 86

#### Chapter 6 Conclusions ............................................. 89

* 6.1 Thesis Contributions .................................................. 90
* 6.2 Future Directions ..................................................... 92

#### References .......................................................... 94

#### Appendices ........................................................... 101

* Appendix A: Efficient Algorithm to solve 2-Variable Linear Programs ... 102
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 4.1</td>
<td>Optimal solution of scheduling problem with 5 routine and 3 add-on customers with different cost ratio $\alpha = c^f/c^w$, $Z_i \sim U[20, 40]$, $d = 190$.</td>
<td>48</td>
</tr>
<tr>
<td>Table 4.2</td>
<td>Optimal time allowances for varying cost ratios for 5 routine and 3 add-on patients with LogNormal service times.</td>
<td>50</td>
</tr>
<tr>
<td>Table 4.3</td>
<td>Computational performance of standard ND, multi-cut version of ND, and the two-variable algorithm implemented within ND.</td>
<td>52</td>
</tr>
<tr>
<td>Table 4.4</td>
<td>Computational performance of Standard ND and Standard ND with (VI-1), (VI-2) and (VI-3) ($Z_i \sim U(20, 40)$).</td>
<td>52</td>
</tr>
<tr>
<td>Table 4.5</td>
<td>Value of the Stochastic Solution for several test instances with $Z_i \sim U[20, 40]$ and $q_i = 0.5$ for add-on requests.</td>
<td>53</td>
</tr>
<tr>
<td>Table 5.1</td>
<td>Optimal Appointment Schedules with FCFS and AOFS for a 9 routine + 3 add-on customers problem with $Z_i \sim 3+ Lognormal(23.55, 11.89)$.</td>
<td>65</td>
</tr>
<tr>
<td>Table 5.2</td>
<td>Second stage variables for a 3 customer problem given a first stage solution for sequencing decisions is known.</td>
<td>73</td>
</tr>
<tr>
<td>Table 5.3</td>
<td>Optimal solution of a 5 customer problem instance with identical patient characteristics, $c^w_i = 4$, $c^a_i = 2$, $p_i = 0.5$, $\forall i$, $c^f = 10$, $c^s = 5$, $Z_i \sim U(20, 40)$, $\forall i$.</td>
<td>78</td>
</tr>
<tr>
<td>Table 5.4</td>
<td>Alternative optimal solution of a 5 customer problem instance with identical patient characteristics, $c^w_i = 4$, $c^a_i = 2$, $p_i = 0.5$, $\forall i$, $c^f = 10$, $c^s = 5$, $Z_i \sim U(20, 40)$, $\forall i$.</td>
<td>78</td>
</tr>
<tr>
<td>Table 5.5</td>
<td>Optimal solution of a 3 Routine + 2 Add-on customer problem instance, $c^w_i = 4$ $\forall i = 1, 2, 3$, $c^w_i = 8$ $\forall i = 4, 5$, $c^a_i = 2$ $\forall i = 1, 2, 3$, $c^a_i = 6$ $\forall i = 4, 5, c^f = 10$, $c^s = 5$, $p_i = 0.5$, $Z_i \sim U(20, 40)$, $\forall i$.</td>
<td>79</td>
</tr>
<tr>
<td>Table 5.6</td>
<td>Optimal solution of a 3 Routine + 2 Add-on customer problem instance, $c^w_i = 4$ $\forall i = 1, 2, 3$, $c^w_i = 8$ $\forall i = 4, 5$, $c^a_i = 2$ $\forall i = 1, 2, 3$, $c^a_i = 4$ $\forall i = 4, 5, c^f = 10$, $c^s = 5$, $p_i = 0.5$, $Z_i \sim U(20, 40)$, $\forall i$.</td>
<td>80</td>
</tr>
<tr>
<td>Table 5.7</td>
<td>Optimal sequencing rules with respect to different cost parameters and conditional probabilities $p$ in the presence of customers with different variances.</td>
<td>81</td>
</tr>
<tr>
<td>Table 5.8</td>
<td>Computational performances of solution methods with uniformly distributed service times ($Z_i \sim U(30, 40)$). Problems were solved to a tolerance of 1% with a maximum time of 15000 seconds. An asterisk (*) indicates cases in which no model instance were solved to the specific tolerance. Times are reported in seconds.</td>
<td>84</td>
</tr>
</tbody>
</table>
Table 5.9  Computational performances of solution methods with Lognormally distributed service times ($Z_i \sim \text{Lognormal}(3.2, 0.5)$). Problems were solved to a tolerance of 1% with a maximum time of 15000 seconds. An asterisk (*) indicates cases in which no model instance were solved to the specific tolerance. Times are reported in seconds.

Table 5.10  L-shaped gap at the time of termination for the instances that are not solved to optimality.

Table 5.11  Results for the problem instances that are solved to optimality. ($c^l = 10, c^u = i, c^w = i^2, Z_i \sim U(30, 40)$ for uniformly distributed service durations, $Z_i \sim \text{LogN}(3.2, 0.5)$ for lognormally distributed service durations.)
LIST OF FIGURES

Figure 2.1 Single server scheduling problem. The scheduler selects start times. The solid line depicts one possible scenario which includes some waiting, idling, and overtime. ................................................................. 6
Figure 2.2 Example of a scenario tree with 10 scenarios over four stages. .......... 18
Figure 3.1 Effects of cost ratio \( \alpha = \frac{c^f}{c^w} \) and no-show probability \( p \) on optimal schedule for 10 customer problem compared to the case in which all customers arrive (\( p = 0 \)), \((Z_i \sim U(20, 40), d = 200)\) .................................................. 32
Figure 3.2 Effects of cost ratio \( \alpha = \frac{c^f}{c^w} \) and no-show probability \( p \) on optimal schedule for 10 customer problem compared to the case in which all customers arrive (\( p = 0 \)), \((Z_i \sim U(20, 40), d = 200)\) .................................................. 32
Figure 3.3 Total waiting time and overtime costs with respect to the number of customers scheduled with different no-show environments, \( Z_i \sim U[20, 40], d = 200, \alpha = 10 \). ........................................... 33
Figure 4.1 Illustration of the scheduling problem with probabilistic arrival of customers given \( n^U = 3 \). .......................................................... 37
Figure 4.2 Tree of scenarios for M-SLP problem with \( n^U \) customers. Solid nodes denote stages in which additional customer appointment requests are pending, and dashed nodes define the day of service given a certain number of customer arrivals. ......................................................... 39
Figure 4.3 Structure of the optimal solution of 5 routine and 3 add-on customers with different cost ratio \( \alpha = \frac{c^f}{c^w} \), \( Z_i \sim U[20, 40], d = 190 \). .......................................... 49
Figure 4.4 Structure of the optimal solution of 5 routine and 3 add-on patients with different cost ratio \( \alpha = \frac{c^f}{c^w} \), \( Z_i \sim U[20, 40], d = 190 \). .......................................... 49
Figure 4.5 Effects of number of add-on customers on the optimal schedule, \( Z_i \sim U[20, 40], d = 250, \alpha = 10 \). .................................................. 51
Figure 4.6 Comparison of convergence for ND, and ND with added (VI-3) (\( n^U = 40 \), \( Z_i \sim U[20, 40], d = 450, q_i = 0.5 \)). .................................................. 53
Figure 5.1 Illustration of the online scheduling problem for scheduling of up to 5 customers. Figure (A) illustrates the case in which the sequence of appointments is FCFS; Figure (B) illustrates the general case in which the FCFS sequence is relaxed. ......................................................... 57
Figure 5.2 Illustration of the sequencing and scheduling problem with probabilistic arrival of 3 customers given AOFS sequence. ................................................. 61
Figure 5.3 Scenario tree for M-SLP problem with 3 customers given that customer 3 one has higher priority. ................................................................. 62
Figure 5.4 Appointment times with varying waiting cost ratios for routine and add-on customers. .......................................................... 66
Chapter 1

Introduction

In many service industries appointment scheduling systems provide the interface between customer demand and service provider availability. Therefore appointment scheduling systems must balance the needs of these two stakeholders. Customers seek to minimize the time they spend waiting for the service, while service providers seek to minimize the amount of time resources are left idle, and the amount of overtime to complete the scheduled services. Whether the scheduling is done for accounting services, consultants, legal services, barber shops/beauty salons, visa services or health care providers, the aim of balancing these conflicting criteria is a common goal.

In this thesis we develop new appointment scheduling models which can be adapted to many of the service systems mentioned above. The research is motivated in part by problems faced by health care systems due to the emerging need for improvement in this industry. In health care delivery systems, achieving a balance between conflicting criteria is particularly important because of the high cost of resources, including human and physical resources. Total health care spending in the U.S. is in 2009 was estimated to reach $2.5 trillion, and it continues to rise at the fastest rate in U.S. history [72]. According to a recent report from the National Academy of Engineering and the Institute of Medicine [71], the health care system is suffering a serious crisis related to high costs and poor access to health care services. The report identifies the design and implementation of better planning and scheduling systems as an important area to study.

Appointment scheduling in the presence of uncertain customer demand is a serious problem that has not yet been well addressed. For instance, at outpatient clinics, failed appointments (customer no-shows) have been observed in the range of 12% to 42%, creating serious inefficiencies [54]. Furthermore, in the clinic and surgery environments most scheduling is done dynamically under uncertainty in the number of patients to be scheduled on a particular day. Nevertheless, outpatient appointment scheduling decisions must be made prior to knowing the
number outpatients that will actually arrive. Scheduling systems that account for uncertain
customer demand can help reduce waiting time for patients and lower costs to the providers.
Reduced waiting time also improves patient satisfaction. In this thesis we present several new
models and new methods to improve efficiency of appointment scheduling systems.

1.1 Research Objective

Scheduling appointments is challenging due to a number of complicating elements. First of
all, the service durations can be affected by various factors depending on the industry and the
nature of the service being provided. Service durations are often highly uncertain, depending
on the requirements of each customer, as well as the differences between each service provider.
In the context of appointment scheduling for surgeries, for instance, the service durations can
be highly uncertain due to varying experience levels of surgeons and operating room (OR)
teams, the presence of residents or surgical fellows in academic medical centers, or patient
characteristics like age, weight, or other unobservable physiological factors that only become
known at the time of the surgery. As a result, simplifying assumptions, such as exponential
service durations, are typically not appropriate.

Another complicating factor in appointment scheduling is the uncertainty associated with
customer arrivals. The exact number of customers to be served on a given day is often not known
at the time of assigning appointments. This arises due to arrivals of additional unanticipated
customers (often called add-ons) or due to no-shows of scheduled customers. No-shows refer
to customers who are assigned to an appointment time but without warning fail to keep their
scheduled appointments on the day of service. Urgent add-on cases, on the other hand, may
result from a sudden and unpredictable need for access to a health service. For instance, in the
surgery scheduling context, urgent add-on cases often arise on short notice. In the outpatient
setting this type of uncertainty occurs due to unexpected arrivals of urgent care patients in
primary and specialty care clinics. Compounding the problem is the fact that appointment
scheduling systems must allocate capacity (time with a provider) dynamically before having
complete knowledge of the number of patients to be scheduled. Reserving capacity for short
notice add-on cases, or compensating for no-shows through overbooking, carries the risk of
over or underutilizing perishable resources with high fixed costs. This combination of dynamic
appointment scheduling, and uncertainty in the load on resources, gives rise to challenging
optimization problems that have not yet been well studied.

In our research, we develop new scheduling models for single server service systems with
uncertain service times and uncertain customer demand. We consider the dynamic nature of ap-
pointment scheduling and provide optimal or near optimal schedules incorporating the presence
of the above mentioned types of uncertainty. Our research also aims to incorporate different
customer types with varying priorities, service duration distributions, and arrival patterns.

1.2 Summary of Research Contributions

Appointment scheduling systems have been widely studied in the past. Many previous studies focus on applying queuing theory principles assuming Markovian behavior and exponential service times. However, these assumptions are not representative of most scheduling environments. Furthermore, such studies assume a stationary and steady state behavior which is unrealistic in most service environments (e.g., hospitals and outpatient clinics operate largely on an 8 – 12 hour day). Our research makes a number of contributions to the existing literature including the following:

- New stochastic programming models that consider two prominent types of arrival uncertainty: no-shows and urgent add-on customers
- A study of the dynamic nature of the appointment scheduling by using a multi-stage stochastic programming formulation
- The first dynamic sequencing and scheduling optimization models for on-line appointment scheduling
- Novel solution methods based on adaptations of several decomposition methods that take advantage of the structure of the proposed models
- Theoretical insights about the optimal appointment schedule based on special cases

To our knowledge, our research is the first attempt to determine the optimal appointment schedules in the presence of uncertain duration and customer demand in a dynamic scheduling environment.

1.3 Organization of the Document

In Chapter 2 we present a detailed literature review on appointment scheduling. We also present a literature review on two-stage and multi-stage stochastic programming to introduce the main methodological concepts employed in this thesis. In Chapters 3 and 4 we present two appointment scheduling models under the assumption of a fixed first-come-first-serve (FCFS) sequence of customer appointments. Chapter 3 presents a static appointment scheduling model in which the number of customers to be scheduled is known in advance. In this model we incorporate customer no-shows into the appointment schedule and investigate the changes in the optimal schedule in the presence of different no-show probabilities. Chapter 4 presents
a dynamic appointment scheduling model in which the total number of customers are not known due to unexpected customer arrivals. The model is a multi-stage stochastic program in which customers are scheduled FCFS. We present several solution methods, computational experiments to compare the methods, results for several realistic test instances, and a case study that provides insights into the structure of the optimal schedule in the context of endoscopy suite scheduling. Chapter 5 presents a new model which relaxes the FCFS sequence assumption and therefore provides a general model for dynamic appointment sequencing and scheduling. Finally, Chapter 6 summarizes our main results, contributions to the literature, and future research directions.
Chapter 2

Literature Review

In this chapter, we begin by reviewing the extensive literature on appointment based scheduling systems that has appeared in both the operations research and the health care literature. We particularly emphasize recent studies that consider new and important aspects of these scheduling systems. We provide a taxonomy of the literature based on the type of operations research methodology used. Finally, we briefly discuss some open challenges and opportunities that exist for future research.

The second part of this chapter is a review of two-stage and multi-stage stochastic programming methods. This is the primary solution methodology used to solve the dynamic appointment scheduling problem described in the following chapters. We discuss some basic concepts about multi-stage stochastic programs, and summarize the main ideas that have been previously proposed to improve the performance of solution methods.

2.1 Appointment Based Scheduling Systems

Appointment scheduling systems arise in the context of many types of service systems. Much of the appointment scheduling literature has focused on a single stochastic server with the objective of finding scheduled start times for the service of each customer. This problem is well known and widely studied in the literature [90] [65] [42] [43]. The aim of the single server scheduling problem is to assign start times for customers to be scheduled for that server on a given day. The scheduler must consider the variability in service durations. Depending on the choice of start times there may be more or less waiting or idling that occurs between customers. Also, depending on the length of time the server is planned to be available, some overtime may occur at the end of the day. Figure 2.1 demonstrates the situations when these conditions occur.

It is commonly assumed that service times are random, and a deterministic schedule of
Figure 2.1: Single server scheduling problem. The scheduler selects start times. The solid line depicts one possible scenario which includes some waiting, idling, and overtime.

Appointment times is selected to optimize competing performance criteria including expected customer waiting time, server idle time and/or overtime. Customers are assumed to be punctual and no recourse is available to the scheduler once customers start to arrive on the day of service (e.g., customers cannot be rescheduled). This problem differs from typical single server queuing models in two important ways. First, the scheduling horizon is assumed to be finite, typically limited by the number of customers seen on a particular day. Second, customers are assumed to arrive deterministically according to a defined schedule of appointment times. Thus, the focus is on the transient behavior under deterministic arrivals, as opposed to steady state and stochastic arrival assumptions that are common in the queuing literature.

There are a number of literature review papers related to appointment scheduling. Cayirli and Veral [15] provide a comprehensive literature survey on appointment scheduling in the context of outpatient services. They present the problem along with the complicating environmental factors such as the number of different services and physicians, uncertainty in service durations, punctuality of patients, no-shows, walk-ins etc. They also present the various performance criteria including mean total cost of waiting time of patients and overtime and idle time of service providers, as well as various time-based, congestion based, productivity based performance criteria used in previous appointment scheduling literature. The authors also provide a detailed taxonomy of the methodologies used in the literature.

More recently Gupta and Denton [39] described some of the challenges that need to be addressed in appointment scheduling in health care systems including primary care, specialty care and surgery scheduling. The authors claim that indirect patient waiting time, which is
the waiting time between the appointment request and the actual appointment date, should be considered in addition to patient waiting time on the day of the service. They also point out a number of open research questions in need of further analysis including the effect of late cancelations and no-shows, patient preferences, and resource allocation depending on different patient types.

A large portion of the appointment scheduling literature has developed around applications to surgery scheduling. This is in part due to the extremely high cost associated with surgery. Early reviews [75] [61] [10] on surgery scheduling focus on classifying the previous research based on various stages of decision making, performance measures such as OR utilization and costs, and the scheduling process. More recent review papers provide detailed classifications based on surgery durations (deterministic or stochastic), patient arrivals (elective and non-elective), and operations research methodology [38] [14] [30].

In the remainder of this section we provide a literature review emphasizing recent studies that consider new and important aspects of appointment based scheduling systems. We categorize the literature based on the type of operations research methodology used.

### 2.1.1 Queuing Models

There have been numerous queuing based studies presented over the past several decades on the problem of scheduling patients for surgeries and outpatient clinic appointments [90] [89] [65] [46] [82] [66]. Queuing research has focused on single server problems for which appointment decisions are economically significant. Scheduling problem contexts include manufacturing lines, transportation, and numerous health service systems. For instance, in [78], the authors consider scheduling the arrival of cargo ships at a seaport. They consider a single server queue with general service distribution and deterministic arrivals (D/G/1 queue). In their treatment of the problem the costs of underutilization of a seaport (server) are traded off against the cost of cargo ship (customer) waiting.

In the queuing literature, authors typically assume an infinite horizon which is not typical in most appointment scheduling environments. Some researchers propose more realistic queuing models with a finite horizon. For instance, Brahimi and Worthington [13] develop a queuing system with time dependent Markovian arrival rate and discrete service time distributions (M(t)/G/s queue) where the number of customers at any time is assumed to be finite. Wang [87] studies the scheduling problem both as a static scheduling problem in which a finite number of customers are scheduled at once, and as a dynamic scheduling problem in which an additional number of customers are scheduled one at a time after an initial batch of customers have been scheduled. He uses phase-type distributions to investigate the transient solution of a Markovian server with general arrival distribution (an S(n)/M/1 queue) to determine optimal start times.
for each customer.

Vanden Bosch and Dietz [11] present a queuing model with phase-type service durations and deterministic arrivals where patient no-shows are also considered. In their study, patients are classified into different groups depending on their service durations. Analyzing the special structure of the model, they propose an algorithm which guarantees to find the optimal schedule along with the optimal sequence efficiently. More recently, Hassin and Mendel [41] develop a queuing model for a single Markovian server to analyze the effects of patient no-shows on customer waiting times, server idle time and overall waiting and idle time costs per customer. Recently, Zeng et al. [94] presented a study on overbooking of clinical appointment schedules to reduce the negative impact of no-shows. The authors developed and studied a queuing model with patients having heterogenous no-show probabilities. They propose that overbooking is beneficial for open access scheduling systems which allow patients to request appointments on the same day of service.

Queuing models can offer valuable general insights about the effect of uncertainty on scheduling decisions. However, they also have several shortcomings, particularly in the context of appointment scheduling in health care systems. For instance, many of the queuing papers cited above assume that the system reaches steady state. However, appointment based service systems typically operate for some specific period of time during the day (e.g. 8-12 hours). Thus, the system terminates service at regular intervals and the steady state assumption is a significant approximation that has not been carefully validated as an approximation. Queuing models also often require strong assumptions such as exponentially distributed service durations, which is not reasonable for most types of service systems, such as surgical operations which tend to better fit a lognormal distribution [83] [64].

2.1.2 Simulation

Simulation models have found considerable application to appointment based scheduling systems. Although they do not provide closed form solutions like queuing models, simulation models are more flexible in terms of assumptions about the probability distributions of service times. However, they are also more computationally intensive than queuing models and may require considerable computation time.

Studies based on discrete event simulation models relax many of the assumptions of the queuing literature. For example, Vissers and Wijngaard [86] construct a model for simulation to find suitable appointment systems for outpatient clinics and propose reductions in the number of variables by incorporating patient no-shows and walk-ins in the distribution of consultation durations. Charnetski [19] was among the first to use simulation in the surgery scheduling context. He used simulation to determine the relationship between the starting times of surgeries
and the costs of two types of idle time, idle time of the surgeon and idle time of the personnel and facilities. He considers a single operation room (OR) where patient arrivals are assumed to be deterministic but surgery durations are random.

Ho and Lau [42] [43] used simulation in their studies to compare the performance of simple scheduling rules in a single server system. They propose that the performance of the scheduling rules are affected by environmental conditions of the operating environment such as the probability of no-shows, number of patients to schedule, and the service distribution. To account for no-shows they consider simple rules such as scheduling 2, 3 or more patients simultaneously at the beginning of the session, and scheduling the later patients with intervals equal to the mean of the service distribution. They also explore modifications of these rules by changing the interval time between patients. In total they evaluated more than 50 different rules with several different combinations of the three environmental conditions. They present the Pareto set of scheduling rules with respect to expected idle time of the server and expected patient waiting time.

Many researchers have used simulation to analyze more complex systems that go beyond the single server assumption such as multi-OR surgical environments with recovery areas. Scmitz and Kwak [79], for example, used simulation to analyze a multi-OR surgical suite with recovery rooms to determine the number of ORs to open on a day given that the number of surgeries to perform is known. They also study the impact of an increase in the number of ORs on recovery room usage and determine the need for recovery room capacity given that some of the surgeries are done in outpatient setting. Lowery [58] simulates patient flow through a hospital’s critical care units including ORs, ICU and PACU beds to determine the bed requirements for these recovery units. Lowery and Davis [59] developed a simulation tool to analyze the effects of changes in surgery durations and the surgery schedule on the number of ORs needed.

Simulation methods have been widely used for testing the performance of scheduling heuristics. Huschka et al. [45] developed a discrete event simulation model for a newly designed outpatient surgical suite to study the impact of scheduling and start time heuristics and daily surgical mix on the expected patient waiting time and overtime. The authors tested the performance of seven combinations of OR allocation heuristics and sequencing rules such as scheduling surgeries according to longest processing time first (LPT), shortest processing time first (SPT), increasing variance (VAR) and increasing covariance (COV) and found that in general LPT and combination of LPT and VAR rules (LPT-VAR) perform better than other rules. The authors also found that arrival time schedules and daily surgery mix have significant effects while scheduling rules have smaller effect on the performance measures. Recently Gul et al. [37] develop a discrete event simulation model to compare the performances of the heuristics considered in [45] with a bi-criteria genetic algorithm which finds near optimal solutions.

Some authors have considered a broader range of decisions including short, medium, and
long range decisions. For example, Testi et al. [84] used simulation in a three-phase surgical suite planning approach. The authors used optimization in the first two phases to assign the available OR time to specialties and to create the Master Surgical Schedule which determines the OR to surgical specialty assignments. In the third step, they evaluate different surgery sequences depending on the longest time spent in the waiting list (LWT), LPT and SPT using simulation. They found that the SPT rule performs better in minimizing the overtime and number of surgeries needing to be postponed to the next day.

2.1.3 Optimization

Optimization techniques have been widely applied to appointment scheduling problems. Most researchers have used deterministic models under the assumption of known service durations and customer demand for scheduling appointments. Some have also used stochastic optimization models in order to incorporate uncertainties related to service durations and customer arrivals. We begin by describing deterministic models and then move on to stochastic optimization models.

In a multi-server scheduling context, Ozkarahan [73] proposed a goal programming model for scheduling surgeries in a surgical suite with multiple operating rooms including multiple objectives such as OR utilization (considering OR idle time and overtime), OR and surgeon preferences, and intensive care capacity. For each of these objectives, a goal is specified and the objective is to minimize the deviations from these goals. Surgeon specific operation durations are assumed to be accurately estimated and considered to be deterministic for this study.

Jebali et al. [47] developed a two-step mixed integer programming (MIP) formulation for surgery scheduling. In their 2-step model, the first step consists of assigning surgeries to ORs. They consider a number of constraints including the OR, surgeon, equipment, and Intensive Care Unit (ICU) bed availability, as well as the due dates of the surgeries for patient satisfaction with the objective of minimizing total cost of overtime and idle time of ORs and patient waiting time. The second step deals with finding the surgery sequences which minimize the overtime in each OR assuming all surgery-to-OR assignment decisions have been made.

Pham and Klinkert [74] extended the job shop scheduling problem to a multi-mode blocking job shop in order to satisfy resource constraints (modes) specific to the surgery environment. They use a MIP model formulation with the objective of minimizing the weighted sum of makespan and the sum of starting times of all procedures. They propose that add-on and emergency surgeries can be scheduled by adding new constraints using job insertion. Job insertion inserts new surgeries into the established schedule and bumps some elective surgeries to the following day if necessary to perform emergent surgeries. In [32], the authors develop a integer programming (IP) model to assign elective surgeries with deterministic durations into multiple
ORs. The objective is to minimize the weighted sum of overutilization and underutilization of ORs. They solve instances optimally with branch-and-price algorithm.

There are also a number of stochastic optimization models for appointment scheduling. The majority assume a single server, and determine optimal start times for each customer’s service (see Figure 2.1 in section 2.1.1). Weiss [88] provides the optimal start times for a simple two surgery problem with uncertain surgery durations in a single OR. Balancing surgeon’s idle time and waiting time, the author shows that the problem is equivalent to the well known newsvendor problem from inventory theory, for which the optimal solution is known in closed form. Weiss also examined the optimal sequence of two cases by looking at the expected cost resulting from the sequences. He showed that sequencing decisions are related to the dispersion of the density function of surgery durations. He proposes that scheduling surgeries that have distributions with “fatter” tails first results in higher total expected costs of waiting and idling due to the impact on the following surgery. Wang [87] used phase-type distributions to obtain close form expressions for expected customer waiting time and server idling. Using these closed form expressions he solves a nonlinear program to compute schedules that minimize a weighted sum of expected waiting and idling.

Denton and Gupta [21] study a general two-stage stochastic linear programming formulation of the single server appointment scheduling problem. They determine the optimal start times considering expected patient waiting time, expected OR idle time, and expected overtime with respect to a defined length of day as the performance criteria. They develop an iterative approximate method based on upper and lower bounds on the optimal solution to the problem with continuously distributed surgery durations. The authors propose that the appointment schedule with random service durations should follow a dome shape indicating that the start times for earlier and later customers need to be closer to each other than the start times in the middle of the schedule. Robinson and Chen [76] studied the same problem using Monte Carlo integration which is based on replacing the true service time distribution with an approximate discrete distribution composed of randomly generated points. The authors also proposed a simple heuristic method which follows the dome shaped optimal structure of the schedule.

Gerchak et al. [34] use a stochastic dynamic programming model for a multi-period surgery scheduling problem. In particular they use a Markov decision process to investigate the optimality of using cut-off policies. Cut-off policies suggest rejecting next elective cases to be able to accept possible emergent surgeries if a certain number of ORs are already occupied by elective cases. The authors found that using these cut-off policies are not always optimal but there is a small difference between the total cost of the system using the cut-off policy and the total cost of the optimal schedule. Lamiri et al. [50] propose a stochastic model for scheduling elective surgeries over a planning horizon, however they consider OR capacity for both elective and add-on patients. They use simulation to generate samples for emergency surgeries. The
samples are used to estimate deterministic capacity requirements for a MIP model that minimizes total overtime costs and other costs associated with hospitalization, and penalties for waiting time of the patients, violating patient’s and surgeon’s preferences and deadlines. Their model is solved via branch-and-bound and then compared to the solution of a simulation optimization approach. The authors conclude that simulation optimization offers improvements even for small sample sizes.

Denton et al. [24] proposed a two-stage stochastic mixed integer programming formulation to find optimal allocation of surgery blocks to ORs. They also develop a robust formulation which aims to minimize the maximum cost (worst-case outcome) that may result from the set of uncertain surgery durations. The robust formulation is advantageous when limited data is available for surgery durations. It is also much less computationally intensive than their stochastic programming model. Batun et al. [1] model a multi-OR scheduling problem as a stochastic MIP to determine the number of ORs to open on a given day, allocation of surgeries to ORs, sequence of surgeries within each OR, and start times for each surgeon. The focus of their study is on parallel scheduling of surgeries in which a surgeon may be allocated two or more ORs on the day of surgery.

Recently, Lamiri et al. [51] and Fei et al. [31] used column generation for scheduling of elective surgeries. In [31], the problem is formulated as a deterministic integer program, while in [51], a random capacity for emergency surgeries is reserved and the problem is formulated as a stochastic integer program. In both cases, the models aim to find the assignments of the surgeries into ORs on a given day in the planning horizon. Column generation is used as a solution technique in which columns represent possible surgery-to-OR assignments in a given planning horizon. The authors then use heuristic methods to find a feasible solution to the integer program and to improve the solution.

Despite the wide literature on optimization methods in appointment based scheduling problems, there is still a lack of literature in optimization of multi-server appointment scheduling. The current literature suggests that considering the stochastic aspects of the surgical environment is essential for solving realistic problems. Developing efficient stochastic optimization methods to solve larger and more realistic problems which consider multiple servers still remains as an open challenge for researchers.

### 2.1.4 Heuristics

The stochastic and combinatorial nature of scheduling problems gives rise to computationally challenging optimization models. Many researchers have addressed this problem by proposing fast heuristics. This enables consideration of more complex and more realistic models. Guinet and Chabaane [36] consider a multi-OR surgical suite with a recovery room. They formulate
a MIP model and propose a primal-dual heuristic to find assignment of patients on a weekly basis. They also propose to revise the schedule (reschedule) daily to accommodate changes in recovery room availability and cleaning operations during the day. In their model they minimize the total cost which includes overtime costs due to late closure of ORs, and patient waiting costs. They consider constraints on OR availability, overtime and surgeons’ capacity. They also include constraints based on surgeon and equipment unavailability.

Denton et al. [23] developed a two-stage stochastic mixed integer programming model of the single OR scheduling problem with the addition of surgery sequencing. Decisions in their model are the optimal start times for surgeries and the optimal sequence of surgeries. An interchange heuristic is used to find near optimal sequences, and start time decisions are computed to optimality for each sequence generated by the interchange heuristic. They also prove for a two-surgery example conditions under which a stochastic ordering defines the optimal sequence, and use numerical experiments to demonstrate that sequencing in order of increasing variance of surgery duration is typically optimal or near optimal. Intuitively this is because delaying highly uncertain surgeries to the end of the day limits their impact on other surgeries in the schedule. Vanden Bosch and Dietz [12] propose a local heuristic method to sequence the customers in a first-come-first-serve appointment system when customers differ by waiting cost, no-show probability and service time distributions. The authors also propose a solution method to find the optimal schedule given the sequence is fixed.

Sier et al. [80] used simulated annealing to compute near optimal solutions to a mixed integer nonlinear programming model. In their model, they schedule surgeries depending on the patient age (younger patients earlier in the day), surgery type and duration by applying the LPT rule and equipment availability by checking conflicts and rescheduling surgeries if necessary. They study a system with four ORs and find schedules that satisfy all of the resource constraints. More recently, Denton et al. [22] combined simulation with a simple simulated annealing heuristic to study the impact of uncertainty in a multi-OR outpatient surgical environment. Beliën and Demeulemeester [2] also propose a simulated annealing approach as well as several different MIP heuristics to create a master surgery schedule for a block-booking system with the objective of minimizing the expected shortage of beds. Their MIP heuristics involve solving a number of MIPs which aim to minimize the maximum expected duration, maximum expected bed occupancy and maximum weighted sum of mean and variance of the daily bed occupancy respectively. Their simulated annealing approach, on the other hand, starts with a random surgery block and does pairwise exchanges of blocks to find an improved schedule with respect to bed shortage.

Bin packing algorithms have been widely used to find the mix of different cases for multi-server environments [27] [26]. In this context the servers are available for a fixed time during the day, and represent the bins. The customers, which have an estimated duration, are the items
to be packed in the bins. There are two types of bin packing problems that have been studied: on-line and off-line. In the on-line bin packing problem customers are scheduled sequentially one at a time. Many heuristics have been proposed for this problem. One well known heuristic, called best fit, schedules customers to the servers which has the least amount of remaining time that is sufficient to fit the case. Another heuristic, called worst fit, schedules the customers into the emptiest server which has enough time to fit the case.

In the off-line version of the bin-packing problem customers are batched and simultaneously assigned to servers. LPT is a well known heuristic for off-line bin packing. To apply LPT, service durations for customers are sorted based on their mean durations and then assigned sequentially to the servers. Many variants of LPT exist including best fit descending in which cases are assigned sequentially to the fullest server after being sorted by LPT rule and worst fit descending in which they are assigned to the emptiest server. Dexter et al. [26] compare several of these heuristics for scheduling add-on surgeries into remaining open OR times. They found that off-line algorithms are able to fit more add-on surgeries and in particular best fit descending with special constraints is more likely to maximize OR utilization.

Houdenhouven et al. [44] leverage the portfolio effect which is a risk management technique used in finance. Portfolio management is used to reduce the risk (minimize variance) or increase the profitability (maximize expected return) by distributing the investments into various different projects instead of investing in a single project [63]. The authors use the portfolio effect with bin packing heuristics to consider uncertainty in surgery durations. They propose to use planned slack time between surgeries depending on the variances of different surgery durations. They conclude that scheduling surgeries with similar duration variability together will reduce the total slack time to allow between surgeries and thus increase available OR time. Hans et al. [40] propose both constructive and local search heuristics to minimize the total planned slack. They found that, as a result of the portfolio effect, surgeries with similar duration variability are often scheduled on the same day for each specialty.

2.2 Literature Review on Stochastic Programming

Most decision making processes involve some form of uncertainty. Optimization of decisions under uncertainty allows the decision maker to make better plans about the future and to hedge against various possible future outcomes. In modeling and solving stochastic problems, however, it is common to replace random variables by their mean. Thus, many real world problems are formulated as deterministic problems where all random parameters are assumed to be known with certainty. These deterministic models often fail to perform well under conditions of uncertainty. Stochastic programming is the branch of mathematical programming that aims to find optimal decisions under conditions of uncertainty.
A shortcoming of stochastic programming is that the problem size tends to grow in proportion to the number of possible realizations (scenarios) of uncertain parameters. The large-scale nature of the stochastic programs therefore creates the need for solution methods that exploit the special structure of these problems. Commonly employed methods include Dantzig-Wolfe decomposition (inner linearization) and Bender’s decomposition (outer linearization) which decompose the large-scale problem into a master problem and several independent subproblems. Dantzig-Wolfe decomposition proceeds by adding new columns to the master problem based on solutions of the subproblems [20]. Benders decomposition proceeds by adding new constraints (supporting hyperplanes known as optimality cuts) that are computed using dual solutions to the subproblems [3]. The remainder of this chapter presents the general formulation of stochastic linear programs with an emphasis on methods that are most relevant to this thesis. We review two-stage and multi-stage stochastic programs and related solution methodology. We also review research on techniques developed to accelerate the performance of the decomposition-based methods.

2.2.1 Stochastic Linear Programs and Solution Methods

The classical stochastic program is the two-stage stochastic linear program (2-SLP) which has first stage and second stage decisions. First stage decisions, $\mathbf{x} \in \mathbb{R}^n$, are made without complete information on the collective outcome of a vector of random variables, $\xi \in \Xi$, where $\Xi$ defines the set of all possible outcomes. Second-stage decisions, $\mathbf{y} \in \mathbb{R}^{m_2}$, also called recourse decisions, are taken in response. The mathematical formulation of 2-SLP is:

$$
\begin{align*}
    z_{SP} &= \min \ c^T \mathbf{x} + E_\omega [Q(\mathbf{x}, \omega)] \\
    \text{s.t} & \quad A \mathbf{x} = \mathbf{b} \\
    & \quad \mathbf{x} \geq 0
\end{align*}
$$

(2.1)

where

$$
Q(\mathbf{x}, \omega) = \min \{ q(\omega)^T \mathbf{y}(\omega) | W(\omega) \mathbf{y}(\omega) = h(\omega) - T(\omega) \mathbf{x}, \mathbf{y}(\omega) \geq 0 \}
$$

(2.2)

denotes the recourse function [8]. The matrix $A \in \mathbb{R}^{m_1 \times n_1}$ and vector $\mathbf{b} \in \mathbb{R}^{m_1}$ define first stage constraints which are known with certainty. The matrices $W \in \mathbb{R}^{m_2 \times n_2}$, $T \in \mathbb{R}^{m_2 \times n_1}$, and the vectors $q \in \mathbb{R}^{m_2}$, and $h \in \mathbb{R}^{m_2}$ depend on random outcome indexed by $\omega \in \Omega$ where $\Omega$ denotes the set of all random outcomes. $E_\omega [\cdot]$ denotes the expectation with respect to $\omega$.

The computational challenge of stochastic programming lies in the number of possible outcomes, also called scenarios. Several approaches exist in the literature to evaluate the value
of solving a stochastic program. One common approach, called *wait and see* (WS), assumes that the uncertainty is resolved (the outcome \( \omega \) is observed) and then the optimal first stage decision, \( x \), is made. This is not realistic since it assumes that the decision maker has perfect information about the random events before the time of the first stage decision. However, the WS solution can be compared to the optimal solution of (2.1) (referred to as SP below) to determine the expected value of perfect information (EVPI). For a minimization problem, the optimal objective value of the WS solution, \( z_{WS} \) is less than the stochastic programming solution, i.e. \( z_{WS} \leq z_{SP} \). The difference between the WS solution and the SP solution is

\[
EVPI = z_{SP} - z_{WS}.
\]

(2.3)

EVPI represents the benefit to the decision maker from perfect information about future outcomes.

In the *mean value* or *expected value* (EV) problem, the random variables are replaced with their expected values, and the SP becomes a deterministic linear program. The optimal solution of the mean value problem, \( \bar{x}(\bar{\xi}) \), is then used to solve the stochastic program for each scenario to obtain objective function values of each scenario problems, \( z(\bar{x}(\bar{\xi}), \xi) \), to see how this mean value solution performs over all different scenarios of the stochastic program. This value, called the expected result (EEV) uses the mean value solution as follows:

\[
z_{EEV} = E_{\xi}[z(\bar{x}(\bar{\xi}), \xi)].
\]

(2.4)

The *value of the stochastic solution* (VSS) represents the cost of ignoring uncertainty in making a decision. It is frequently used as an estimate of the benefit of solving a stochastic program. The VSS is defined as follows:

\[
VSS = z_{EEV} - z_{SP}
\]

(2.5)

In general, the larger the VSS, the more important it becomes to solve the SP relative to the solution to the mean value problem.

The scenario based structure of the SP makes decomposition methods attractive. The basic idea of decomposition methods to solve the stochastic programs is using outer linearization. Van Slyke and Wets [81] extends Bender’s decomposition to solve two-stage stochastic linear programs via a method called the *L-Shaped Method*. The L-shaped method is based on dividing the stochastic program into a *master problem* and independent scenario subproblems, one for each scenario \( \omega \). The recourse function, \( Q(x) \), is replaced by a surrogate variable, \( \theta \), in the master problem. The subproblems are solved and their dual solutions are used to generate *feasibility cuts* and *optimality cuts* for the master problem. Together the master problem and
the optimality cuts form an outer linearization of (2.1). The corresponding master problem and subproblems are as follows:

**Master Problem:**

\[
\begin{align*}
\min\quad & z = cx + \theta \\
\text{s.t.}\quad & Ax = b \\
& D_\ell x \geq d_\ell \quad \ell = 1, \ldots, r \quad \text{(Feasibility cuts)} \quad (2.6) \\
& E_\ell x + \theta \geq e_\ell \quad \ell = 1, \ldots, s \quad \text{(Optimality cuts)} \quad (2.7) \\
& x \geq 0, \theta \in \mathbb{R}
\end{align*}
\]

**Subproblems:**

\[
\begin{align*}
\min\quad & w = q^T(\omega) y \\
\text{s.t.}\quad & Wy = h(\omega) - T(\omega)x \\
& y \geq 0
\end{align*}
\]

Initially, solving the master problem without any feasibility and optimality cuts, generates an initial solution \( x_0 \). Using this solution all subproblems are solved individually, and the corresponding dual solutions are used to generate the feasibility and optimality cuts iteratively. Feasibility cuts address the occurrence of infeasible subproblems for a given trial solution \( x^\nu \) at iteration \( \nu \) of the L-shaped method. For some problems, every feasible first stage solution \( x \) has a feasible completion in the second stage. These kind of problems are said to have complete recourse and they do not require feasibility cuts. Optimality cuts are supporting hyperplanes of \( Q(x) \). As optimality cuts are sequentially added to the master problem new trial solutions are obtained. The new trial solutions are passed to the second-stage subproblems. The algorithm iterates until an optimality criterion is met. Convergence to the optimal solution is guaranteed for relatively weak assumptions, given a finite number of scenarios.

### 2.2.2 Multi-stage Stochastic Linear Programs

Many real life problems require a series of decisions to be made periodically over time. At each stage new decisions are made before future realizations of the random elements are known. These types of problems can be formulated as multi-stage stochastic programs (M-SLP). It is assumed that at each stage a finite number of realizations of the future outcomes are possible. Figure 2.2 shows an example of the tree structure for a multistage stochastic program with 4 periods. Each scenario at a certain stage has a single ancestor scenario and one or more descendant scenarios. In this example there are a total of 10 scenarios at stage 4.
Multi-stage stochastic programming has been widely used in modeling multi-period financial planning problems to develop a hedging strategy with the optimum set of investments. There have also been applications to energy, transportation, telecommunications and various other industries [6]. The general formulation of M-SLP with fixed recourse is as follows:

\[
\begin{align*}
\min \quad & z = c^1 x^1 + E_{\xi^2} \left[ \min \{ c^2(\omega^2)x^2(\omega^2) + \ldots + E_{\xi^H} \left[ \min \{ c^H(\omega^H)x^H(\omega^H) \} \ldots \right] \right] \\
\text{s.t.} \quad & W^1 x^1 = h^1 \\
& T^1(\omega^1)x^1 + W^2 x^2(\omega^2) = h^2(\omega^2) \\
& \vdots \\
& T^{H-1}(\omega^{H-1})x^{H-1}(\omega^{H-1}) + W^H x^H(\omega^H) = h^H(\omega^H) \\
& x^1 \geq 0, x^t(\omega^t) \geq 0, t = 2, \ldots, H
\end{align*}
\] (2.9)

where \( t = 1, \ldots, H \) is the index of stages. First stage vectors \( c^1 \) and \( h^1 \) are known, and \( c^t(\omega^t), \ldots, H \).
\( h_t(\omega_t), T_t(\omega_t^{t-1}) \) are scenario dependent in the future stages.

A common method for solving problems in the form of (2.9)-(2.12) is nested decomposition. It is a generalization of the L-shaped method to more than two stages. The nested decomposition method has its roots in deterministic linear programming for problems with a nested constraint structure which is often observed in problems involving dynamic decision making.

In the stochastic programming context it was first proposed by Louveaux [57] for multi-stage quadratic programs and by Birge [5] for multi-stage linear programs. The algorithm generates cuts for the ancestor scenario problem for decision \( x_t \) which has feasible completion in all descendant scenarios. Similar to the L-Shaped method, the nested decomposition achieves outer linearization by iterating backward and forward, passing information between the ancestor master problems and descendant scenario problems iteratively generating feasibility and optimality cuts until it converges to an optimal solution.

A number of different strategies have developed to select the next subproblem to solve during iterations of the nested decomposition method. One such approach was proposed by Wittrock [92] for deterministic problems. In his strategy, the algorithm moves from the first stage problem to the last stage sequentially (provided no infeasibility is detected) and then moves backwards, adding optimality cuts to ancestor stages sequentially until it reaches the the first stage. This strategy is called fast-forward-fast-back. Other strategies that have been tested are fast-back, which forces the algorithm to move from stage \( t \) to \( t - 1 \) whenever possible, and fast-forward, which forces the algorithm to move from stage \( t \) to \( t + 1 \) whenever possible. Gassmann [33] conducted a series of numerical experiments and concluded that fast-forward-fast-back typically performs better than other strategies.

### 2.2.3 Methods to Accelerate Convergence

Decomposition methods such as L-shaped method and nested decomposition can suffer from computational inefficiencies for several reasons. One reason is that the master problem is a poor approximation of the recourse function at the beginning of the solution process. This causes the algorithm to oscillate between solutions that are far from optimal, often generating poor optimality cuts at early stages of the algorithm. Ruszczynski [77] proposed adding a quadratic regularizing penalty term in the objective function which aims to prevent this inefficient oscillation. The penalty term keeps the solution from moving far from the best solution found so far in early iterations. However this comes at the expense of having to solve a quadratic master problem. An alternative approach proposed by Linderoth and Wright [55] adds box-shaped trust regions to the subproblems of the M-SLP. In their approach, at an iteration \( \nu \), bounding constraints in the form of \( -\Delta e \leq x - x' \leq \Delta e \) are added to the problem, where \( \Delta \) is the trust region radius, and \( x' \) is the current iterate. At each iteration \( \nu \), several minor iterations are
performed with different values of $\Delta$ to get a better solution in the next stage, $x^{t+1}$.

Birge and Louveaux [9] propose a different approach to improve the solution performance of the 2-SLPs. The authors present the *multicut L-shaped algorithm* which generates several optimality cuts at each iteration. Instead of aggregating the dual multipliers from each realization into one single cut, one cut per scenario is added to the master problem. Thus more information is provided to the master problem with the goal of decreasing the total number of iterations. However, this is done at the expense of a larger first stage problem.

Another factor which may cause inefficiency in decomposition procedures is the complexity of the subproblems that need to be solved repeatedly at each iteration. Several authors propose methods that exploit the special structure of the subproblems and efficient ways to solve them. Magnanti and Wong [62] consider improving the performance of Bender’s decomposition for solving MIPs by selecting stronger optimality cuts. They point out that the method is a tradeoff between decreasing the number of iterations and adding more optimality cuts which makes the master problem more difficult to solve. The authors propose to use Pareto optimal cuts which are not dominated by other cuts. In addition, they propose that the special structure of the master problem can be taken into consideration to generate better optimality cuts. For example, if the master problem has a network, or a traveling sales person problem (TSP) structure, valid inequalities that are known to be efficient for these special problem structure can be added at the beginning of the solution procedure to minimize the number of iterations. Wentges [91] provides an example of this. He applied Bender’s decomposition for the capacitated facility location problem (CFLP) which has location decisions in the first stage and demand allocation decisions in the second stage. The subproblems in the second stage are defined as transportation problems. The highly degenerate subproblems result in multiple dual optimal solutions thus creating a series of potential optimality cuts. The author suggests that choosing the right dual solutions will produce stronger cuts, and improve convergence.

Laporte et. al. developed new valid inequalities for the stochastic vehicle routing problem (SVRP) and the probabilistic traveling salesman problem (PTSP)[52], [53]. For the SVRP, a new constraint set is developed to limit the excess duration of the route by using expectations of the stochastic routing and service times. For the PTSP, the master problem is relaxed by eliminating the integrality constraints and subtour elimination constraints. Valid inequality and lower bounding functionals on expected tour length are added to the master problem. The authors show that adding these constraints and valid inequalities improves overall convergence.

### 2.3 Specific Contributions to the Literature

Despite the significant literature on optimization methods in appointment based scheduling problems, there is still a lack of literature in optimization of appointment scheduling under
uncertainty. Recent research focuses on problems with uncertain service duration but with little or no emphasis on uncertain customer demand. Furthermore, most work on appointment scheduling is on static scheduling, which assumes that the number of customers to be scheduled is known in advance. In reality scheduling is often dynamic because the number of customers to be scheduled is unknown at the beginning of the scheduling process. Developing efficient stochastic optimization methods to solve these larger and more realistic problems still remains as an open challenge.

The majority of the research to improve the efficiency of the multi-stage stochastic programs (M-SLP) is on parallel processing which is based on solving subproblems in parallel [77] [55] [7]. However, there is a sparse literature on other efficiency improvement techniques for M-SLP. Furthermore, there has been less work on modeling applications with M-SLPs which often provides a more realistic representation of dynamic decision making processes. Most published research considers models with a small number of stages (e.g., 3−5 stages). In contrast we study problems with as many as 30 stages.

In Chapter 3 we present a new 2-SLP model for the appointment scheduling problem in the presence of customer no-shows. No-shows have significant negative effects on the performance of health care systems. Thus, we consider this aspect of uncertain customer arrivals and investigate the structure of the optimal appointment schedule. Our results indicate that, in the presence of no-shows, the optimal appointment schedule includes double-booking of customers. We also derive a sufficient condition for double-booking to be optimal.

In Chapter 4 we present our implementation of nested decomposition to solve a dynamic appointment scheduling problem that we formulate as an M-SLP. A fast algorithm is employed to solve the resulting master problems at each stage. We also extend the multicut implementation of the L-shaped decomposition method proposed by Birge and Louveaux [9] to nested decomposition. Several new valid inequalities are also developed and added to the master problems to improve the convergence of the nested decomposition. A series of computational experiments are performed to evaluate the alternative methods, and to derive insights into the structure of the optimal appointment schedules.

In Chapter 5 we generalize the dynamic appointment scheduling problem by relaxing the assumption of a FCFS sequence. We formulate the problem as a stochastic mixed integer program. For this model, we use a novel approach to maintain non-anticipativity that allows the multi-stage problem to be formulated as a two-stage stochastic mixed integer program with binary decision variables in the first stage. We present two alternative formulations of the problem. We discuss the structure of the problem and a tailored implementation of the integer L-shaped method. We experiment with the addition of valid inequalities and methods for solving the master problem including various strategies for branch and bound method. Finally, we present the results of numerical experiments that provide insights into computational efficiency.
of algorithmic improvements and the structure of the optimal schedules.
Chapter 3

Appointment Scheduling in the Presence of No-Shows

In this chapter we propose a new model for appointment scheduling under uncertainty. In this model, we assume customers may fail to show up (no-show) at their assigned appointment time, with some known probability. This is motivated by the common occurrence of no-shows in outpatient health care environments [54]. In outpatient clinics no-shows have been reported to range from 12% to 42%, making efficient management of resources in outpatient clinics difficult [28] [67].

No-shows not only cause loss of revenue for health care organizations, they also increase uncertainty and bring unforseen costs due to the time reserved but left unused. It is one of the most common problems in scheduling clinic appointments. Health care providers may try to overcome this problem by adapting their scheduling policies. One common approach is to allocate the same appointment time to two or more patients which is often referred as double-booking. Double-booking is commonly used by health care providers to compensate for the idle time caused by no-shows, but at the expense of higher waiting times for the customers that show up.

No-shows have recently attracted the attention of several researchers in both medical and operations research fields. Early research focused on investigating the most common causes of no-shows, and measuring the effects of no-shows on the health care organizations. Deyo et al. [28] reported that several factors may influence the no-show behavior of patients. Such factors include demographics of patients (age, sex, education, income level, socioeconomic level, family size), features and severity of the disease and the treatment, ease of access (distance, cost), and time between the appointment request and the day of the scheduled appointment. In their study, several interventions such as phone reminders and increased patient-doctor interaction are mentioned as potential ways to decrease no-shows. Moore et al. [67] studied the effects of
no-shows on the financial performance of a family medicine clinic. They found that even though 61% of no-show appointment slots were replaced by walk-in patients, the revenue loss due to broken appointments may range from 3% to 14% over a year.

More recently Muthuraman et al. [70] and Zeng et al. [94] proposed overbooking of patients in order to compensate for the no-shows. The authors state that overbooking might cause overloading of the clinic and increase the patient waiting time and provider overtime if too many patients show-up. Higher patient waiting times might then increase patient no-show rates. Thus, they provide insights into the use of overbooking policies under different no-show rates and for different patient types. In both of these studies, service times are considered to be exponentially distributed. Later Chakraborty et al. [18] extended their work by considering general service distributions.

In another study by Liu et al. [56], no-shows are considered in a dynamic multi-day appointment scheduling problem. In this study, the authors proposed solutions to the appointment scheduling problem by assigning the appointment of a patient to a future date depending on the clinic’s schedule at the time of the appointment request by the patient. In other words, the aim is to find the appointment day of the patient as opposed to finding the appointment slot in the studies mentioned above. In this study, the problem is modeled as a Markov decision process. Due to the large state space, instead of determining the optimal policy, several alternative heuristic policies are developed and compared using a simulation study.

Our work differs from the previous research in several ways. Unlike previous research, we are interested in determining the effects of no-shows on the daily optimal appointment schedule. Our focus is on exact methods to determine optimal schedules, rather than heuristics. We formulate a two-stage stochastic linear program (2-SLP) for the appointment scheduling problem with a fixed number of customers with a certain probability of no-show. Unlike previous research which applies queuing principles, our model is not limited to exponentially distributed service durations. We discuss the structural properties of our model and we use L-shaped method described in Chapter 2 to compute the optimal appointment schedules. We present numerical results to get insights about the structure of the optimal appointment schedule in the presence of no-shows. We provide theoretical insights into the optimality of double-booking decisions. Finally, we investigate the changes in the optimal appointment schedule with varying no-show rates and cost parameters.

The remainder of this chapter is organized as follows: In Section 3.1, a model formulation for the appointment scheduling problem in the presence of no-shows is presented. Section 3.2 discusses the structure of the model, and the methodology used to solve the problem. In Section 3.3 we present the results of our computational experiments. Finally, in section 3.4 we summarize general insights and conclusions.
3.1 Model Formulation

The problem addressed here is that of finding the optimal arrival times for \( n \) customers to visit a stochastic server. Service times are assumed to be random variables and the objective is to minimize a weighted sum of expected customer waiting time and expected overtime with respect to an established session length, \( d \). Customers, \( i = 1, \ldots, n \), have no-show probabilities, \( p_i \). We use the following additional notation, where upper case indicates random variables and bold face is used to denote vectors:

**Model Parameters:**

- \( n \): number of customers to be scheduled
- \( \omega \): index for service duration and no-show scenarios
- \( A(\omega) \): random vector of indicators for customer arrival (1) or no-show (0)
- \( Z(\omega) \): vector of random service durations for \( n \) customers
- \( d \): session length to complete all customers before overtime is occurred
- \( c^w \): vector of waiting time cost coefficients for \( n \) customers
- \( c^\ell \): cost coefficient for overtime

**Decision variables:**

- \( x \): vector of time allowances (inter-arrival times) for the first \( n \) customers
- \( w(\omega) \): vector of customer waiting times
- \( s(\omega) \): vector of server idle times between consecutive customers
- \( \ell(\omega) \): overtime with respect to session length \( d \)

The vector of time allowances \( x \in \mathbb{R}^{n-1} \), denotes first stage decisions made in advance of the observation of random service durations and no-shows (note that \( x \) is \( n - 1 \) dimensional since \( x_i \) denotes inter-arrival times between the \( n \) customers). The scheduled appointment time for customer \( i \) is the sum of job allowances from 1 to \( i - 1 \). Thus, customer 1 arrives at time 0, customer 2 at time \( x_1 \), customer 3 at time \( x_1 + x_2 \), and so on. The random service time durations vector, \( Z(\omega) \), has support \( \Xi \in \mathbb{R}^n \), and the possible collective outcomes of service times (scenarios) are indexed by \( \omega \in \Omega \). The vectors \( w(\omega), s(\omega) \in \mathbb{R}^n, \ell(\omega) \in \mathbb{R} \), denote the
second stage (recourse) decisions made after the observation \( \omega \) of random service durations. The parameters \( c^w, c^s, c^\ell \in \mathbb{R} \) denote the cost per unit time for waiting, idling, and overtime, respectively.

Commonly considered criteria for determining optimal time allowances include customer waiting time, server idle time, and overtime, which can be written as follows:

\[
\begin{align*}
  w_i(\omega) &= (w_{i-1}(\omega) + Z_{i-1}(\omega) - x_{i-1})^+, \quad i = 2, \ldots, n, \\
  s_i(\omega) &= (-w_{i-1}(\omega) - Z_{i-1}(\omega) + x_{i-1})^+, \quad i = 2, \ldots, n, \\
  \ell(\omega) &= (w_n(\omega) + Z_n(\omega) + \sum_{i=1}^{n-1} x_i - d)^+ 
\end{align*}
\]

where \((\cdot)^+\) indicates \( \max(\cdot, 0) \). The waiting time and server idle time associated with the first customer is zero \( (w_1(\omega) = s_1(\omega) = 0, \forall \omega) \). i.e. the first customer receives service as soon as they arrive. The optimal appointment schedule is defined by the following unconstrained minimization problem:

\[
\min_{x} \left\{ \sum_{i=1}^{n} c^w E_{\omega}[w_i(\omega)] + c^\ell E_{\omega}[\ell(\omega)] \right\}. 
\]

Some authors have considered a weighted sum of expected server idle time, overtime, and customer waiting time as the objective function; for simplicity, we consider only customer waiting time and overtime as in (3.4). This is justified by the following proposition which is adapted from [21].

**Proposition 1** When \( d = 0 \) expected idle time is equal to expected overtime minus expected total service time:

\[
\sum_{i=1}^{n} E[S_i] = E[L] - \sum_{i=1}^{n} \mu_i 
\]

**Proof:** See [21].

By Proposition 1, our elimination of the idle time is done without loss of generality, since idle time can be considered by setting session length \( d = 0 \).

Formulation (3.4) can be modified to account for no-shows as follows. Define random service durations as \( \hat{Z}_i = A_i(\omega)Z_i(\omega) \) where

\[
A_i(\omega) = \begin{cases} 
0 & \text{with probability } p_i \\
1 & \text{with probability } 1 - p_i.
\end{cases}
\]

\( \hat{Z}_i(\omega) \) is a random variable representing the service duration for customer \( i \) conditional on the customer showing up for their appointment, which occurs with probability \( 1 - p_i \). In
general there is no closed form expression for the solution to (3.4). Denton and Gupta [21] discuss the properties of an equivalent 2-SLP formulation that can be used to achieve significant computational advantages. Similarly, our model can also be formulated as a 2-SLP as follows:

\[
\min \ E_\omega \left[ \sum_{i=2}^{n} c_i^w A_i(\omega) w_i(\omega) + c^\ell(\omega) \right] \\
\text{s.t.} \quad \begin{cases} 
  w_2(\omega) - w_2(\omega) + w_3(\omega) & \geq \hat{Z}_1(\omega) - x_1, \forall \omega \\
  \vdots & \\
  - w_{n-1}(\omega) + w_n(\omega) - w_n(\omega) + \ell(\omega) & \geq \hat{Z}_{n-1}(\omega) - x_{n-1}, \forall \omega \\
  - w_n(\omega) + \ell(\omega) & \geq \hat{Z}_n(\omega) + \sum_{i=1}^{n-1} x_i - d, \forall \omega \\
\end{cases}
\]

\[ x \geq 0, \ w(\omega), \ell(\omega) \geq 0, \forall \omega. \]

We refer to the above model as the no-show appointment scheduling problem (NS-ASP).

### 3.2 Structural Properties and Solution Methodology

Because of the potentially large size of the stochastic program we propose, taking advantage of the problem structure is important. In this section we first provide a special condition on the optimal scheduling of 2 customers in the presence of no-shows, then we present some structural properties of (NS-ASP) and the solution methodology.

#### 3.2.1 No-Shows and Double-Booking

As mentioned in the introduction, double-booking, i.e. the scheduling of two or more customer arrivals simultaneously, is a commonly used approach in practice. In this section, we investigate a special case for which double-booking is provably an optimal policy. Specifically, we consider the special case of \( n = 2 \) and \( d = 0 \). This instance of (NS-ASP) is equivalent to the well known newsvendor problem:

\[
\min_{x_1} \{ E_\omega [c^w(Z(\omega) - x_1)^+ + c^\ell(x_1 - Z(\omega))^+] \}.
\]

Taking the derivative of the cost function \( E_\omega [c^w(Z(\omega) - x_1)^+ + c^\ell(x_1 - Z(\omega))^+] \) and setting the
derivative to 0, we get the following expression:

\[ e^t F(Z(\omega) < x_1) - e^w F(Z(\omega) > x_1) = 0 \]  \hspace{1cm} (3.6)

\[ e^t F(Z(\omega) < x_1) - e^w (1 - F(x_1 > Z(\omega))) = 0, \]  \hspace{1cm} (3.7)

where \( F(\cdot) \) is the cumulative distribution function (cdf) of \( Z(\omega) \). From equations (3.6) and (3.7), it follows that

\[ F(Z(\omega) < x_1) = \frac{e^w}{e^w + e^t}. \]  \hspace{1cm} (3.8)

There is at least one \( x_1 \) satisfying (3.8), and the optimal solution is:

\[ x_1^* = \min \{ x_1 \geq 0 : F(x_1) \geq \frac{e^w}{e^w + e^t} \}. \]  \hspace{1cm} (3.9)

this can be expressed alternatively as follows: \( x_1^* = F^{-1}\left( \frac{e^w}{e^w + e^t} \right) \).

In the case of no-shows, we replace the random variable for service time, \( Z(\omega) \), with

\[ \hat{Z} = A(\omega)Z(\omega) \]  \hspace{1cm} (3.10)

where

\[ A(\omega) = \begin{cases} 0 & \text{with probability } p \\ 1 & \text{with probability } 1 - p. \end{cases} \]

We now prove the following sufficient condition for optimality of double-booking customers 1 and 2.

**Proposition 2** The optimal schedule for \( n = 2, d = 0 \), with no-show probability \( p \), will double-book customers 1 and 2 if \((\frac{e^w}{e^w + e^t}) \leq p\).

**Proof:** The optimal solution is given by

\[ x_1^* = \min \{ x_1 \geq 0 \mid F(x_1) \geq \frac{e^w}{e^w + e^t} \}. \]  \hspace{1cm} (3.11)

Note that double-booking in this case corresponds to \( x_1 = 0 \), i.e. customers 1 and 2 both arrive at time 0. Since \( \hat{Z}_i \) is 0 with probability \( p \), the cumulative distribution function (cdf) of \( \hat{Z}_i \), \( F_{\hat{Z}_i}(0) = p \). The optimal solution to (3.11) is \( x_1^* = 0 \) since \( F_{\hat{Z}_i}(0) = p \) and by assumption \( \frac{e^w}{e^w + e^t} \leq p \). \( \square \)

Proposition 2 indicates that the two customers should be double-booked when the no-show probability is above a certain level, \( \frac{e^w}{e^w + e^t} \). This provides insight into the occurrence of double-
booking in optimal appointment schedules in the presence of no-shows. In the Section 3.3 we provide results supporting the optimality of double-booking for larger problems with \( n > 2 \) when no-shows are present.

### 3.2.2 Subproblem Structure

The first stage problem in (NS-ASP) consists of first stage variables and nonnegativity constraints, and can be written as follows:

\[
\begin{align*}
\min \{ Q(x) \}, \\
x \geq 0
\end{align*}
\]

where \( Q(x) = E_{\xi}[Q(x, \xi(\omega))] \) is the recourse function representing the expected total cost of all scenario subproblems.

The second stage problem consists of separate scenario subproblems \( Q(x, \xi(\omega)) \). Each subproblem includes an objective function which minimizes the total costs of waiting and overtime, and constraints to compute the waiting time and overtime variables. A scenario subproblem for (NS-ASP) can be written as follows:

\[
Q(x, \xi(\omega)) = \min \sum_{i=2}^{n} c_i^w A_i(\omega)w_i(\omega) + c^\ell \ell(\omega)
\]

\[
s.t. \quad w_2(\omega) \geq \hat{Z}_1(\omega) - x_1, \forall \omega \\
-w_2(\omega) + w_3(\omega) \geq \hat{Z}_2(\omega) - x_2, \forall \omega \\
\quad \vdots \\
-w_{n-1}(\omega) + w_n(\omega) \geq \hat{Z}_{n-1}(\omega) - x_{n-1}, \forall \omega \\
-w_n(\omega) + \ell(\omega) \geq \hat{Z}_n(\omega) + \sum_{i=1}^{n-1} x_i - d, \forall \omega
\]

\[
x \geq 0, \ w(\omega), \ell(\omega) \geq 0, \forall \omega.
\]

Note that (NS-ASP) has complete recourse since the second stage is feasible for any \( x \in \mathbb{R}^{n-1} \) (see 2.2.1 in Chapter 2). That is, for feasible allowance decisions \( x \), found at the first stage, the second stage waiting time, and overtime variables will always be feasible. The solutions to scenario subproblem (3.13) in (NS-ASP) are trivial since the waiting time and overtime variables can be found easily without solving a linear program. Instead they can be determined by evaluating the corresponding waiting and overtime functions. Similarly, the dual solutions...
can be computed efficiently using a backward recursion procedure as follows:

\[
\pi_i(x, \omega) = \begin{cases} 
0 & w_i(\omega) = 0, \\
c_i^{w_{i+1}} + \pi_{i+1}(x, \omega) & w_i(\omega) > 0,
\end{cases}
\]

for \(i = 1, \ldots, n - 1\) and

\[
\pi_n(x, \omega) = \begin{cases} 
0 & \ell(\omega) = 0, \\
c^\ell & \ell(\omega) > 0.
\end{cases}
\]

For complete information about subproblem solutions please see [21].

### 3.2.3 L-Shaped Method

(NS-ASP) model is solved using the L-shaped method which, as discussed in Chapter 2, is a standard decomposition method to solve 2-SLPs. The L-shaped method proceeds by solving a master problem and scenario subproblems iteratively. The master problem consists of first stage variables, constraints and a surrogate variable, \(\theta\), which represents the recourse function. At each iteration, \(\nu\), of the method, a set of first stage allowance decisions, \(x^\nu\), is found by solving the relaxation of the master problem and then send to the scenario subproblems. Using this first stage solution, waiting time decision variables, \(w^\nu\), and overtime decision variable, \(\ell^\nu\), are computed for each scenario subproblem. The dual solutions of each subproblems computed using (3.14) and (3.15) are then used to generate an optimality cut which will be added to the master problem. The algorithm terminates iterating when a solution which cannot be improved by adding new optimality cuts. The steps of L-shaped method is summarized as follows:

**L-Shaped Algorithm**

1. \(\nu = 1, \ k = 1\)
2. Start with an arbitrary solution \(x\)
3. While (current bound - \(\theta\) > tolerance) do
   4. \(\nu \leftarrow \nu + 1\)
   5. Solve master problem (3.12)
   6. Solve subproblem (3.13) for each \(k\)
   7. Add optimality cut to master problem
4. end while
3.3 Results

In this section we provide the results of numerical experiments to illustrate the structure of optimal schedules in the presence of no-shows. All experiments are done with 10,000 randomly generated service duration scenarios, which we have found sufficient to achieve tight confidence intervals on the optimal solution. The methods proposed in Section 4.2 were implemented in C++ with the CPLEX 11.0 callable library to solve the master problems. All experiments were done on a Intel Core2Quad CPU, Q6600 GHz 2.39 GHz, 3.25GB RAM.

Figure 3.1 illustrates the optimal allowances for a 10 customer problem for varying no-show probabilities ($p = 0$, $p = 0.2$, $p = 0.3$) when cost ratio $\alpha = c^f/c^w = 0.1$ and $\alpha = 1$. When no-shows are not present ($p_i = 0 \forall i$), the optimal schedule exhibits a dome shape, i.e., shorter allowances for patients early and late in the session, and larger allowances for the patients in the middle. This pattern has been observed for static scheduling problems [21]. In the presence of no-shows, with probability $p = 0.2$, and $p = 0.3$, the optimal allowances are reduced and customers are scheduled within shorter intervals. The reduction in appointment intervals is consistent with hedging against high idling that is caused by customers who do not show up. Thus, the average overtime is lower when some customers no-show.

Figure 3.2 presents the results of experiments with increasing $\alpha$ values. Note that, without the presence of no-shows, i.e. $p = 0$, the dome shape is preserved regardless of the value of $\alpha$. However, this pattern breaks down as the value of $\alpha$ and probability of no-show increase. When $\alpha = 10$, we observe double-booking for the first two customers, i.e. the first two customers are scheduled to arrive at time 0. As $\alpha$ increases, consistent with intuition, we observe additional double-bookings for the customers early in the schedule. For $\alpha = 10000$ and $\alpha = 1000$, customers 1, 2, 3, and 4 are double-booked, for $\alpha = 100$, customers 1, 2 and 3 are double-booked, and for $\alpha = 10$, customers 1 and 2 are double-booked. It is interesting to note that all double-booking occurs at the beginning of the day, independent of no-show probability and cost parameters. It is also worth mentioning that significant double-booking only occurs for very high overtime costs, even when no-show probabilities are high.

Figure 3.3 depicts the total waiting time and overtime costs with respect to the scheduled number of customers with and without the presence of no-shows. Note that the total cost decreases as the probability of no-show increases. This is a natural result of having lower expected waiting time and overtime when there are fewer customers in the system, in the presence of no-shows. The negative effects of no-shows can be observed when the expected number of customers scheduled is considered. For instance, when the total number of customers is 10, the expected number of customers scheduled is 8 when no-shows occur with probability 0.2. In this case 8 customers are scheduled with a total cost of 708.31. However, 8 customers could be scheduled more efficiently with a total cost of 511.54 if no-show probabilities would
Figure 3.1: Effects of cost ratio $\alpha = \frac{c^f}{c^w}$ and no-show probability $p$ on optimal schedule for 10 customer problem compared to the case in which all customers arrive ($p = 0$), ($Z_i \sim U(20, 40), d = 200$)

be reduced to zero. Thus, in this example, no-shows cause a 38% increase in cost to the scheduling system. Similarly, when no-shows occur with probability $p = 0.3$, the expected value of customers scheduled is 7 and the total cost decreases to 448.91. On the other hand, 7 customers could be scheduled with a total cost of 198.38 if no-shows probabilities could be
reduced to zero. In this case, the optimal schedule in the presence of no-shows is 126% more costly than the optimal schedule without no-shows. It is also worth noting that the total cost is nonlinearly increasing in the number of customers; however it is nearly linear for \( n \geq 8 \). This can be interpreted as resulting from the high probability of overtime as the number of customers in the system increases.

![Figure 3.3: Total waiting time and overtime costs with respect to the number of customers scheduled with different no-show environments, \( Z_i \sim U[20, 40], d = 200, \alpha = 10 \).](image)

### 3.4 Conclusions

In this chapter we presented a 2-SLP model for appointment scheduling in the presence of no-shows. We modified the model provided by an earlier work, [21], by incorporating customer no-shows. We present sufficient conditions for double-booking for the special case of \( n = 2 \) customers. We also investigated the effects of no-shows on the daily optimal schedules and on the total cost of the scheduling system.

Our results exhibit a dome-shape, such as previously observed in static scheduling problems, in the absence of no-shows [21]. The presence of no-shows generally causes the optimal inter-arrival times (allowances) between customer appointments to decrease. Our findings indicate that as the probability of no-show increases, it may be optimal to double-book some customers. Double-booking is common in practice and our results show that it is also optimal in some
cases where overtime or idling costs are high, or no-show probabilities are high. In general, we observe double-booking increases with no-show probability. Double-booking is typically optimal only when overtime costs are higher than waiting time costs (e.g. a factor of 10 or more). Furthermore it is optimal to double-book customers at the beginning of the schedule.

The negative effects of no-shows on the total cost of the scheduling system can be observed when the expected number of customers in the system is considered. Our results indicate that even though the total cost appears to be decreasing with no-shows, the total cost of the schedule increases when the actual number of customers in the system is taken into account. In the presence of no-shows, the expected number of customers in the system decreases, thus, fewer number of customers are served often at a substantially higher cost. For instance, we found that, for a specific 10 customer instance, scheduling 8 customers is 38% more costly when no-shows occur with \( p = 0.2 \). This percentage increases to 126% when the no-show rate increases to 0.3. Thus, expected costs appear to be highly sensitive to no-show probabilities.

In conclusion, we observed that the percentage of the extra cost due to the uncertainty caused by no-shows increases significantly as the no-show probability increases. In other words, as the no-show probability decreases, more customers can be scheduled at the same or lower cost. These results provide motivation for implementation of methods for reducing customer no-shows (e.g. financial penalties for the no-show customers, automated appointment reminders etc.)
Chapter 4

Dynamic Appointment Scheduling with Uncertain Demand

In Chapter 3 we introduced a static appointment scheduling model which assumes that the number of customers is known in advance. However, in many service systems appointment scheduling is complicated by the fact that the exact number to be scheduled for a particular day is not known with certainty until the day of service. Instead, customers request appointments sequentially over time, and appointments are quoted on-line. Since rescheduling of appointments is uncommon in most service industries it is necessary to make these on-line scheduling decisions in such a way that schedules are adaptable to variation in customer demand. Our study is motivated in part by problems faced by health care providers who schedule a nominal number of appointments in advance of a given day, and then must accommodate some high priority patients that may arrive on short notice. A number of other applications of this problem have been identified in the literature including material handling, scheduling cargo ports and outpatient services. It is frequently the case that appointments are quoted dynamically to customers with imperfect knowledge of total demand.

In health care delivery systems, uncertainty in demand arises due to the inherently uncertain nature of urgent care. In outpatient clinics, for instance, customers are often classified into groups such as routine and urgent. Routine patients are scheduled in advance, often weeks or months in advance. Urgent patients are scheduled on much shorter notice, typically days in advance. Therefore, when routine appointment scheduling must be done in a way that anticipates the potential future need to schedule additional urgent patients.

The dynamic nature of on-line appointment scheduling has been recognized as a challenge by health care professionals. In surgery delivery systems, for instance, urgent add-on cases arise on short notice and create the need to dynamically update schedules to accommodate these high priority cases [35, 25]. One commonly used policy is known as the carve out model. This
process reserves some portion of each day in advance for urgent care requests that may arise in the future. Selecting the amount of time to carve out is challenging. Reserving too little results in delayed patient access, while reserving too much much results in wasted capacity.

More recently, the advanced access model has been proposed [69, 68]. The theme of advanced access is to “do today’s work today”. Under this model patient’s calling to see their physician are given an appointment on the same day. Since schedulers do not know exactly how many patients will be scheduled during any given day, appointments must be quoted on-line in such a way that decisions hedge against various possible scenarios.

Other outpatient practices, such as specialty care providers also face similar challenges in appointment scheduling. For instance, a colonoscopy is a standard procedure used to screen patients for colo-rectal cancer. Colonoscopy is also used to diagnose patients that present with symptoms of colo-rectal cancer or other diseases of the gastrointestinal system. The former, screening patients, are schedules well in advance, while latter patients are scheduled quickly on short notice. Similar examples arise in most specialty care practices including neurology, endocrinology, and cardiology. Applications also arise in hospital based surgery practices in which urgent add-on cases arise on short notice prior to a particular day of surgery.

In this chapter we present a new stochastic programming model for dynamic scheduling of appointments to a single stochastic server. The criteria are similar to the classical appointment scheduling problem; however, dynamic nature of appointment scheduling over time necessitates a multi-stage model with scenarios that represent service time uncertainty, and uncertainty in the number of appointment requests. The model is a generalizable representation of the appointment scheduling process for many kinds of service systems. We assume customers request appointments sequentially, one at a time, prior to the day of service. We refer to this problem as the dynamic appointment scheduling problem. Upon request, customers are quoted a particular arrival time during the day. On the day of service customers arrive at their appointed times and receive service with uncertain durations. Thus, waiting time, server idle time, and overtime depend on the outcome of service durations, and the design of the appointment schedule.

4.1 Dynamic Appointment Scheduling

The model presented in this section assumes customers are scheduled dynamically as they call to request an appointment. Appointment requests are probabilistic, i.e., the total number to be scheduled is not known with certainty, and there is a maximum of $n_U$ customers that will be scheduled ($n_U$ as an upper bound on the capacity of the system). The probability of an appointment request by customer $i$, conditional on the appointment request by customer $i-1$, is $q_i$. In other words, $q_i$ exists only if customer $i-1$ is scheduled. We assume customers are
scheduled first-come-first-serve (FCFS) in the sequence of their appointment requests.

### 4.1.1 Simple Examples

To illustrate the nature of our problem we consider two simple examples.

**Example 1** \((n^U = 2, q_2 = 1)\): This represents the case where 2 customers will be scheduled with certainty. From Proposition 1 when \(d=0\) this corresponds to the newsvendor problem.

**Example 2** \((n^U = 3, q_1 > 0, q_2 > 0)\): In this case one customer will certainly be scheduled. With conditional probabilities \(q_2\) and \(q_3\) customers 2 and 3 may request appointments. For this example there are three customer arrival scenarios:

1. The first customer is scheduled. The 2\(^{nd}\) and 3\(^{rd}\) customers do not request appointments.
2. The 2\(^{nd}\) customer requests an appointment after the 1\(^{st}\) customer is scheduled. The 3\(^{rd}\) customer does not request an appointment.
3. The 2\(^{nd}\) customer requests an appointment after the 1\(^{st}\) customer is scheduled. The 3\(^{rd}\) customer requests an appointment after the 2\(^{nd}\) customer is scheduled.

The sequential nature of the uncertainty in the customer requests in Example 2 is illustrated in Figure 4.1. In contrast to Example 1, a closed form expression for the solution to this problem is not easily obtained.

Figure 4.1: Illustration of the scheduling problem with probabilistic arrival of customers given \(n^U = 3\).

In Example 2 uncertainty is resolved sequentially as appointment requests arise and appointments must be scheduled with imperfect information about the number of customers and their
service times. Therefore each request is treated as an additional stage in the decision making
process. At each stage, \( j \), the time allowance decision, \( x_j \), is made for customer \( j \), without
perfect knowledge of the number of additional future appointment requests. To formulate our
model we use similar notation to that of (NS-ASP), with an additional index, \( j = 1, ..., n^U \) to
denote the stage. Thus, \( w_{j,i}(\omega) \) is the waiting time of the \( i \)th customer on the day of the service,
given \( j \) customers request appointments. We let \( \omega_j \) index service duration scenarios for stage
\( j \). Similarly, we let \( \ell_j \) denote the overtime given \( j \) customers are scheduled. Thus for Example
2, customer arrival scenario 1 and service duration scenario \( \omega_1 \) can be written as:

\[
\begin{align*}
  w_{1,1}(\omega_1) &= 0 \\
  \ell_1(\omega_1) &= (Z_1(\omega_1) - d)^+.
\end{align*}
\]  

For customer arrival scenario 2 and service duration scenario \( \omega_2 \):

\[
\begin{align*}
  w_{2,1}(\omega_2) &= 0 \\
  w_{2,2}(\omega_2) &= (Z_1(\omega_2) - x_1)^+ \\
  \ell_2(\omega_2) &= (x_1 + w_{2,2}(\omega_2) + Z_2(\omega_2) - d)^+.
\end{align*}
\]  

For customer arrival scenario 3 and service duration scenario \( \omega_3 \):

\[
\begin{align*}
  w_{3,1}(\omega_3) &= 0 \\
  w_{3,2}(\omega_3) &= (Z_1(\omega_3) - x_1)^+ \\
  w_{3,3}(\omega_3) &= (w_{3,2}(\omega_3) + Z_2(\omega_3) - x_2)^+ \\
  \ell_3(\omega_3) &= (w_{3,3}(\omega_3) + Z_3(\omega_3) + x_1 + x_2 - d)^+.
\end{align*}
\]  

Thus the three arrival schedules define the waiting time and overtime associated with one, two,
and three scheduled customers, respectively. The indices \( \omega_1 \in \Omega_1, \omega_2 \in \Omega_2, \omega_3 \in \Omega_3 \) refer
to service time scenarios given one, two, and three customers are scheduled, respectively.

4.1.2 Model Formulation

The appointment request scenarios for the dynamic appointment scheduling process are rep-
resented by the tree in Figure 4.2. In the figure, nodes represent the number of customers in
the system. Solid nodes denote the number of scheduled customers and a state in which the
schedule is waiting for future appointment requests. The dashed nodes define the day of service
given a certain number of customers that have requested appointments. Our model starts with
2 customers since the solution of the 1 customer problem is trivial. Starting with 2 customers,
customer 3 requests an appointment with probability \( q_3 \), and with probability \( 1 - q_3 \) no ad-
ditional customers are scheduled. Given a third customer requests an appointment, a fourth customer will request an appointment with probability $q_4$, and so on.

Figure 4.2: Tree of scenarios for M-SLP problem with $n^U$ customers. Solid nodes denote stages in which additional customer appointment requests are pending, and dashed nodes define the day of service given a certain number of customer arrivals.

We formulate this model as the following unconstrained optimization problem:

$$
\min_{x_1} \{ (1 - q_3)Q_1(x_1) + \min_{x_2} \{ q_3(1 - q_4)Q_2(x_2) + \cdots + \min_{x_{n^U-1}} \{ (\prod_{i=3}^{n^U}(q_i)Q_{n^U-1}(x_{n^U-1}) \} \} } \} \quad (4.4)
$$

where $Q_j(x_j) = E_{\omega_j}[Q_j(x_j, \omega_j)]$ denotes the expected cost given that $j + 1$ customers request appointments (note that $Q_j(x_j)$ corresponds to $j + 1$ customers since $x_j$ is the inter-arrival time between customers $j$ and $j + 1$). We refer to $Q_j(x_j, \omega_j)$ as the terminal subproblem for stage $j$ under service duration scenario $\omega_j$ (represented by dashed nodes in Figure 4.2). We refer to this as terminal since it represents the case in which no additional customers beyond $j + 1$ request appointments. Although not explicitly denoted in the formulation it is implied that decision $x_j$ is made prior to knowledge of whether customer $j + 1$ (or additional customers) will request an appointment. We use this implicit definition to simplify the notation, rather than explicitly write a series of linking constraints between stages. The terminal subproblem for stage $j$ can be written as:
\[ Q_j(x_j, \omega_j) = \min_{w, s, \ell} \left\{ \sum_{i=2}^{j+1} c_i^w w_{j,i}(\omega_j) + c^\ell_{j+1}(\omega_j) \right\} \]

s.t
\[ w_{j,2}(\omega_j) \geq Z_1(\omega_j) - x_1 \]
\[ -w_{j,2}(\omega_j) + w_{j,3}(\omega_j) \geq Z_2(\omega_j) - x_2 \]
\[ \vdots \]
\[ -w_{j,j}(\omega_j) + w_{j,j+1}(\omega_j) \geq Z_j(\omega_j) - x_j \]
\[ -w_{j,j+1}(\omega_j) + \ell_{j+1}(\omega_j) \geq Z_{j+1}(\omega_j) + \sum_{i=1}^{j} x_i - d \]
\[ w_{j,i}(\omega_j) \geq 0 \, \forall i, \ell(\omega_j) \geq 0. \]

Formulation (4.4) can be formulated recursively, with \( R_j(x_j) \) denoting the expected cost-to-go given that additional customers may arrive as follows:

\[ R_j(x_j) = \min_{x_{j+1}} \{(1 - q_{j+2})Q_j(x_j) + q_{j+2}R_{j+1}(x_{j+1})\}. \]  

The recursion terminates at the last stage, \( n^U - 1 \), with \( R_{n^U - 1}(x_{n^U - 1}) = Q_{n^U - 1}(x_{n^U - 1}) \). Thus (4.4) can be expressed as:

\[ \min_{x_1} R(x_1). \]  

We refer to the above M-SLP as the dynamic appointment scheduling problem (D-ASP). We discuss several ways to take advantage of the structure of the recursive structure of (D-ASP) in Section 4.2.

It is worth noting some special cases of (D-ASP) which correspond to specific applications. First, in some applications it may be appropriate to assume a certain minimum number of customers arrive with certainty. This is equivalent to defining a lower bound on the number of customers that will be scheduled. This is motivated by health care applications such as hospital based colonoscopy practices where a certain minimum number of patients are scheduled in advance (outpatients), and some uncertain number of urgent add-on cases are scheduled on short notice (inpatients). This is also representative of a common primary care appointment scheduling system called advanced access [69], in which some patients are booked in advance and some urgent patients call for appointments on the day they want to be seen.

For simplicity, in (D-ASP) we have not considered no-shows, however, (NS-ASP) and (D-ASP) could easily be integrated to include the possibility of no-shows which are common in both of the dynamic scheduling applications described above. It is also worth noting that (D-
ASP) is a special case of (NS-ASP) where \( q_i = 1 \) and \( Z_i \)'s are distributed as in (3.2.1). Thus, some of the methods we develop to take advantage of the structure of (D-ASP) are also directly applicable to (NS-ASP) and the standard static appointment scheduling problem [21].

### 4.2 Structural Properties and Solution Methodology

Nested decomposition is a common approach to take advantage of the recursive structure of M-SLPs [5]. It is based on outer linearization of the recourse function, \( R_j(x_j) \), at each stage \( j \). At each stage a solution, \( x_j \), is generated by solving a relaxed master problem which is a linear program representing the expected waiting and overtime cost for scheduled customers, and the expected cost-to-go for future stages. At each (terminal) stage, subproblems based on a number of service time scenarios indexed by \( \omega_j \) are solved given the solution to the master problem, \( x_j \). The dual solutions to the subproblems are used to generate supporting hyperplanes (called optimality cuts) for the stage \( j \) recourse function. These cuts are added sequentially at each stage until the relaxed master problem converges to the optimal solution. The following subsections describe a number of opportunities to improve efficiency of the nested decomposition method (ND) for formulation (D-ASP).

#### 4.2.1 Subproblem Structure

The dual solution to terminal subproblems, \( Q_j(x_j, \omega_j) \), can be computed efficiently using the following backward recursion:

\[
\pi_{j,i}(x_j, \omega_j) = \begin{cases} 
0 & w_{j,i+1}(\omega_j) = 0, \\
\ell_i^w + \pi_{j,i+1}(x_j, \omega_j) & w_{j,i+1}(\omega_j) > 0,
\end{cases}
\]

for \( i = 1, ..., j - 1 \) and

\[
\pi_{j,j}(x_j, \omega_j) = \begin{cases} 
0 & \ell_j(\omega_j) = 0, \\
\ell_j^e & \ell_j(\omega_j) > 0.
\end{cases}
\]

(4.7)
(4.8)

This closed form expression for the dual allows efficient generation of optimality cuts at each stage of the ND algorithm [21]. The master problems of each stage (except the last stage) includes another subproblem that represents the expected cost-to-go, \( R_j(x_j) \) for the remaining future stages. The master problem is based on the following equivalent (outer linearization) formulation:
\[ \min \{ \theta_j \mid \theta_j \geq R_j(x_j) \} \]  
(4.9)
in which the decision variables are \( x_j \) and \( \theta_j \). Thus the master problem for stage \( j \) is a two variable linear program (LP) with optimality cuts at stage \( j \) of the form:

\[ \theta_j + E_j x_j \geq e_j - \sum_{k=1}^{j-1} E_k x_k. \]

Substituting the known values for \( x_1, x_2, \ldots, x_{j-1} \), determined in stages 1, 2, \ldots, \( j \), the cut then takes the following general form for each stage, \( j \), at each iteration, \( \nu \), of the nested decomposition method:

\[ \theta_j \geq \alpha_{\nu} x_j + \beta_{\nu} \]  
(4.10)

and the master problem at iteration \( \nu \) has the following general form:

\[ \min \{ \theta_j \mid \theta_j \geq \alpha_k x_j + \beta_k, k = 1, \ldots, \nu \}. \]  
(4.11)

A linear time method was developed by Dyer [29] to solve two-variable LPs with this special structure. We adapt the algorithm to incorporate nonnegativity constraints on the decision variables (the algorithm is summarized in the Appendix).

### 4.2.2 Valid Inequalities

The standard ND method is attractive for (D-ASP) given the special structure of the subproblems discussed in Section 4.2.1. However, slow convergence of outer linearization methods such as this has been observed by several authors (see, for example, [62]). This results from the fact that little information is available in the form of optimality cuts at early stages of the algorithm, and significant degeneracy in subproblems results from the outer linearization process [5]. We examine opportunities to overcome this problem using lower bounding inequalities based on the mean value problem.

Batun et al. [1] first used the mean value problem to generate valid inequalities for accelerating convergence of the L-shaped method for two-stage stochastic programs. We propose some variants of these valid inequalities that are suited to our M-SLP formulation. The valid inequalities are derived from the mean value problem using Jensen's Inequality, \( Q(x_j) \geq Q(x_j, \mu_j) \) [48]. Thus, \( \theta \geq Q(x_j, \mu_j) \) is a valid inequality that can be added to the master problems at stage \( j \). We begin by providing the following property of the mean value solution to (D-ASP) that is central to the development of our valid inequalities.
Proposition 3 The optimal solution to the mean value problem for (D-ASP) is $\overline{x}_i = \mu_i$.

Proof: Replacing all random variables, $Z_i$, $i = 1, 2, \ldots, n^U$, in (4.6) with their mean, $\mu_i$, it is straightforward to show that $x_i = \mu_i$ results in $w_{i,j} = 0, \forall i$. Since $w_{i,j} \geq 0$ then clearly $x_i = \mu_i$ minimizes $w_{i,j}$. Furthermore, $x_i = \mu_i$ results in overtime $\ell_{j+1} = \sum_{i=1}^{j+1} \mu_i - d$. Substituting $w_{j,2} = (\mu_1 - x_1)$ in (4.5) gives the lower bound $w_{j,3} \geq \mu_1 - x_1 + \mu_2 - x_2$. Following the same substitutions for all $w_{j,i}$ and finally for $\ell_{j+1}$ we obtain the lower bound $\ell_{j+1} \geq \sum_{i=1}^{j+1} \mu_i - d$. Thus $x_i = \mu_i$ simultaneously achieves lower bounds on $w_{j,i}$ and $\ell_{j+1}$, and is therefore the optimal solution to the mean value problem.

We denote the objective function for the mean value problem for stage $j$ at the optimum as $\overline{R}_j(\mu)$. From Proposition 3, $\overline{x}_i = \mu_i, \forall i$ minimizes the stage $j$ mean value problem. Therefore, in the absence of uncertainty in service durations it is optimal to allocate $\mu_i$ to customer $i$, independent of whether there is uncertainty in the number of arrivals.

Lemma 1 The following is a relaxation of $Q_j(x_j, \omega_j)$ where $\omega_j$ denotes the duration scenario for stage $j$:

$$\sum_{i=2}^{j} c_i^w(Z_i(\omega_j) - x_i)^+ + c_j^l(Z_{j+1}(\omega_j) + \sum_{i=1}^{j} x_i - d)^+$$

(4.12)

Proof: From (3.1):

$$w_{j,i}(\omega_j) = (w_{j,i-1}(\omega_j) + Z_{i-1}(\omega_j) - x_{i-1})^+, i = 2, \ldots, j + 1$$

(4.13)

$$\geq (Z_{i-1}(\omega_j) - x_{i-1})^+, i = 2, \ldots, j + 1.$$ (4.14)

From (3.3):

$$\ell_j(\omega_j) = (w_{j,j}(\omega_j) + Z_j(\omega_j) + \sum_{i=1}^{j} x_i - d)^+$$

(4.15)

$$\geq (Z_j(\omega_j) + \sum_{i=1}^{j} x_i - d)^+$$ (4.16)

which completes the proof.

We now use Proposition 3 and Lemma 1 to develop valid inequalities for (D-ASP).

Proposition 4 The following is a valid inequality (VI-1) for outer linearization of (4.6):

$$\theta_j \geq (1 - q_{j+2})(c^l - c_{j+1}^w)x_j + k_1(\mu_1, \ldots, \mu_{j-1})$$ (VI-1)

where $k_1(\mu_1, \ldots, \mu_{j-1}) = (1 - q_{j+2})c^l(\mu_{i+1} + \sum_{i=1}^{j-1} \mu_i - d) + q_{j+2}\overline{R}_{j+1}(\mu)$ is a constant.
Proof: The following is a lower bound on $R_j(x_j)$:

$$R_j(x_j) \geq (1 - q_{j+2})(\sum_{i=1}^{j} c_{i+1}^w (\mu_i - x_i)^+ + c^\ell (\mu_{j+1} + \sum_{i=1}^{j} x_i - d)^+) + q_{j+2} \overline{R}_{j+1}(\mu)$$ (4.17)

which follows directly from Lemma 1, Jensen’s inequality, and substitution of $\overline{R}_{j+1}(\mu)$ into (4.6). Replacing all decision variables $x_i$ with $\mu_i$ in (4.17) excluding the current stage decision variable $x_j$ will result in following lower bound:

$$R_j(x_j) \geq (1 - q_{j+2})(c_{j+1}^w (\mu_j - x_j)^+ + c^\ell (\mu_{j+1} + \sum_{i=1}^{j-1} \mu_i + x_j - d)^+) + q_{j+2} \overline{R}_{j+1}(\mu)$$ (4.18)

Relaxing the nonnegativity functions in the first two terms of the right hand side in (4.18), we obtain the following:

$$\theta_j \geq (1 - q_{j+2})(c^\ell - c_{j+1}^w) x_j + (1 - q_{j+2}) c^\ell (\mu_{j+1} + \sum_{i=1}^{j-1} \mu_i - d) + q_{j+2} \overline{R}_{j+1}(\mu)$$ (4.19)

Note that $k_1(\mu_1, ..., \mu_{j-1})$ in Proposition 4 is a constant, and thus (VI-1) is a linear constraint in two variables $x_j$ and $\theta$. We note the importance of this in preserving the two variable master problem structure discussed in Section 4.2.1.

Proposition 5 The following is a set of valid inequalities for outer linearization of (D-ASP):

$$\theta_j \geq (1 - q_{j+2})(c^\ell - c_{j+1}^w) x_j + (1 - q_{j+2}) c^\ell (\mu_{j+1} + \sum_{i=1}^{j-1} \mu_i - d) + q_{j+2} \overline{R}_{j+1}(\mu)$$ (VI-2)

$$\hat{w} \geq \mu_j - x_j$$

$$\hat{\ell} \geq \mu_{j+1} + \sum_{i=1}^{j} x_i - d$$

$$\hat{w} \geq 0, \hat{\ell} \geq 0$$

where $k_2(\mu_1, ..., \mu_{j-1}) = q_{j+2} \overline{R}_{j+1}(\mu)$.

Proof: The proof follows from Proposition 3 and Lemma 1 and the use of two new variables $\hat{w}, \hat{\ell}$ to linearize the first and second terms in (4.17) which correspond to the waiting time of the last customer and the overtime at stage $j$ respectively. □

Note that $k_2(\mu_1, ..., \mu_{j-1})$ in Proposition 5 is a constant, and (VI-2) is a linear set of constraints in three decision variables $\hat{w}, \hat{\ell}, x_j$. Following is the final valid inequality based on the mean value problem.
Proposition 6 The following is a set of valid inequalities for outer linearization of (D-ASP):

\[
\begin{align*}
\theta_j & \geq (1 - q_{j+2}) \left( \sum_{i=2}^{j+1} c^w \hat{w}_i + c^\ell \hat{\ell}_j \right) + q_{j+2} (1 - q_{j+3}) \left( \sum_{i=2}^{j+2} c^w \hat{w}_i + c^\ell \hat{\ell}_{j+1} \right) \\
& \quad + \cdots + \prod_{k=j+2}^{n_U} q_k \left( \sum_{i=2}^{n_U} c^w \hat{w}_i + c^\ell \hat{\ell}_{n_U - 1} \right) \\
\hat{w}_{i+1} & \geq \hat{w}_i + \mu_i - x_i \quad \forall i \leq j \\
\hat{w}_{i+1} & \geq \hat{w}_i + \mu_i - \hat{x}_i \quad \forall i > j \\
\hat{\ell}_j & \geq \mu_{j+1} + \sum_{i=1}^{j} x_i - d \\
\hat{\ell}_k & \geq \mu_{k+1} + \sum_{i=1}^{j} x_i + \sum_{i=j+1}^{n_U - 1} \hat{x}_i - d \quad \forall k = j+1, \ldots, n_U - 1 \\
x_i & \geq 0 \quad \forall i = 1, \ldots, j, \hat{x}_i \geq 0 \quad \forall i = j+1, \ldots, n_U - 1, \\
\hat{w}_i & \geq 0 \quad \forall i = 2, \ldots, j + 1, \hat{\ell}_k \geq 0 \quad \forall k = j+1, \ldots, n_U - 1
\end{align*}
\]

Proof: (VI-3) follows directly from Proposition 1 of [1] based on adding several new auxiliary variables \(\hat{x}_i, \hat{w}_i,\) and \(\hat{\ell}_k\) that define the mean value problem. The first constraint follows from the fact that the objective of the mean value problem is a lower bound on the optimal solution from Jensen’s inequality. We let \(\hat{w}_i\) denote the waiting time for customer \(i\) in the mean value problem, \(\hat{\ell}_k\) the overtime in the mean value problem, and \(\hat{x}_i\) the time allowance for customer \(i\) in the mean value problem. □

Since (VI-1), (VI-2), and (VI-3) are based on progressively weaker relaxations of the mean value problem they are increasingly stronger valid inequalities. However, the number of constraints in each set is increasing, causing greater computational effort in solving the master problem at each stage of ND. Furthermore, (VI-1) includes only two variables and therefore retains the computational advantage of a two-variable master problem at each stage.

4.2.3 Multi-Cut Outer Linearization:

We use a 2-cut adaptation of the multi-cut L-shaped method proposed by Birge and Louveaux [9]. Based on the structure of (D-ASP) we generate one cut for each of the two terms in the objective function, i.e., the terminal subproblem and the expected cost-to-go. Thus the outer
linearization problem is of the form:

\[
\min (1 - q_j + 2) \theta_j^1 + (q_j + 2) \theta_j^2 \quad (4.20)
\]

s.t. \( \theta_j^1 \geq Q_j (x_j) \quad (4.21) \)

\( \theta_j^2 \geq \min_{x_{j+1}} \{ R_{j+1} (x_{j+1}) \} \quad (4.22) \)

where the right hand side of the cuts are replaced with supporting hyperplanes. In other words we separately outer linearize the right hand side of the two terms in (4.6) using two variables \( \theta_j^1 \) and \( \theta_j^2 \) at each stage \( j \). Thus, we add two optimality cuts to the master problems simultaneously at each iteration.

4.2.4 Nested Decomposition

The ND algorithm proceeds by iteratively improving the approximation of each stage’s convex objective function by adding supporting hyperplanes. Master problems at each stage approximate the expected value of all future stages. (NS-ASP) and (D-ASP) both have complete recourse; therefore, decisions made in a given stage have feasible completion in future stages. Thus, we do not need to consider feasibility cuts in our implementation. In summary we propose the following opportunities to improve efficiency of the ND algorithm: (a) addition of valid inequalities (b) a fast method for solving 2-variable LPs and (c) multi-cut outer linearization. The various implementations of our algorithm are summarized as follows:

**Nested Decomposition Algorithm**

1. \( \nu = 1, j = 1, k = 1 \)
2. Start with an arbitrary solution \( x_j \)
3. While (current bound \( j - \theta_j \) > tolerance) do
   4. Direction \( \leftarrow \) Forward
   5. \( \nu \leftarrow \nu + 1 \)
   6. for \( j = 1 \) to \( n_U - 1 \)
   7. if Valid inequality=True
      8. Add valid inequality (VI-1), (VI-2), or (VI-3) to the master problem
   9. end if
10. Solve master problem \( j \)
11. Solve subproblem \( j \) for each \( k \)
12. end for
13. Direction \( \leftarrow \) Backward

46
14. $\nu \leftarrow \nu + 1$
15. for $j = n^U - 1$ to 1
16. if Standard ND=True
17. Add single optimality cut (4.10) to master problem
18. else (Multi-cut ND=True)
19. Add optimality cuts (4.21),(4.22) to the master problem
20. end if
21. Solve master problem
22. end for
23. end while

In our implementation, master problems were solved with either CPLEX 11.0 or with our implementation of the two-variable LP algorithm of Dyer [29]. Note that the two-variable algorithm cannot be used for the multi-cut ND procedure since it has three decision variables ($\theta_1^j$, $\theta_2^j$, $x_j$). It can only be used in combination with (VI-1) since it is the only set of valid inequalities that maintains the 2-variable structure of the master problems.

The ND algorithm is implemented using the fast-forward-fast-back strategy proposed by [92] which explores all scenarios at stage $j$ before moving forward to stage $j+1$ or backward to stage $j-1$. That is, starting from the first stage, all problems at future stages are solved sequentially as the information gathered from solved problems are passed to the future stages. Upon reaching the last stage, the direction is reversed, and optimality cuts are added to the master problems as each stage. The cycle repeats until no new cuts can be generated. Motivation for the efficiency of this particular strategy is provided by Gassmann [33].

### 4.3 Results

In this section we provide the results of numerical experiments to illustrate the structure of optimal schedules and to evaluate the proposed methods. All experiments are done with 10,000 randomly generated service duration scenarios, which we have found sufficient to achieve tight confidence intervals on the optimal solution. The methods proposed in Section 4.2 were implemented in C++ with the CPLEX 11.0 callable library (except where noted) to solve the linear subproblems and master problems. All experiments were done on a Intel Core2Quad CPU, Q6600 GHz 2.39 GHz, 3.25GB RAM.

We present the results of a series of numerical experiments illustrating the solution time for the various methods proposed, as well as relevant insights related to the value of the stochastic solution (VSS), and sensitivity of the optimal solution to model parameters. We first present the
structure of the optimal appointment schedule and then numerical experiments to evaluate the performance of our algorithms. Next, we compute the VSS for a series of randomly generated model instances. Finally, we present the results of a case study based on a real problem faced at Mayo Clinic in Rochester, MN. We use the case study to illustrate insights about the optimal solution to (D-ASP).

4.3.1 Structure of the Optimal solution

We begin by presenting a simple example that illustrates the structure of the optimal schedule with respect to changes in relative cost of waiting, $c^w$, and overtime, $c^\ell$. The example is motivated by problems typically faced by primary care clinics. In this particular instance, we schedule 5 “routine” customers with certainty, and 3 additional “add-on” customers may request appointments stochastically. The arrival probabilities of the 3 add-on customers are 0.8, 0.5 and 0.3 respectively. The service durations are uniformly distributed between 20 and 40 minutes. Table 4.1 includes the optimal allowance decisions of the problem with different cost ratios.

Table 4.1: Optimal solution of scheduling problem with 5 routine and 3 add-on customers with different cost ratio $\alpha = c^\ell/c^w$, $Z_i \sim U[20, 40]$, $d = 190$.

<table>
<thead>
<tr>
<th>Allowances</th>
<th>$\alpha = 10$</th>
<th>$\alpha = 1$</th>
<th>$\alpha = 0.1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>25.38</td>
<td>32.69</td>
<td>38.52</td>
</tr>
<tr>
<td>$x_2$</td>
<td>30.23</td>
<td>34.14</td>
<td>38.71</td>
</tr>
<tr>
<td>$x_3$</td>
<td>30.65</td>
<td>33.94</td>
<td>38.55</td>
</tr>
<tr>
<td>$x_4$</td>
<td>30.40</td>
<td>33.81</td>
<td>38.57</td>
</tr>
<tr>
<td>$x_5$</td>
<td>29.28</td>
<td>32.59</td>
<td>38.31</td>
</tr>
<tr>
<td>$x_6$</td>
<td>26.28</td>
<td>32.60</td>
<td>38.28</td>
</tr>
<tr>
<td>$x_7$</td>
<td>28.65</td>
<td>32.51</td>
<td>38.28</td>
</tr>
</tbody>
</table>

The optimal structures of the allowance decisions with different cost ratios are depicted in Picture 4.3. The structure of the optimal allowance decisions for routine customers forms a dome shape which requires closer start times for the customers earlier and later in the session and larger allowances for the customers in the middle of the session in order to prevent the schedule to run overtime. This dome shape has been observed for the static scheduling problem in [21]. When the cost ratio decreases from 10 to 1, the allowances between scheduled start times increases since the overtime cost is not higher than waiting time cost. For the cost ratio equals to 0.1, the customers are scheduled with nearly identical inter-arrival times.
Figure 4.3: Structure of the optimal solution of 5 routine and 3 add-on customers with different cost ratio $\alpha = c^\ell / c^w$. $Z_i \sim U[20, 40]$, $d = 190$.

We used our dynamic appointment scheduling model (D-ASP) to evaluate the optimal schedule for an endoscopy suite at Mayo clinic, in Rochester, MN. Endoscopy procedure durations are reported to have a shifted lognormal distribution $3 + \text{Lognormal}(23.55, 11.89)$ by Berg et al. [4]. Based on the analysis of a historical data set for a 6 month period during 2006, 5 routine patients are scheduled for colonoscopy for a given session prior to the day of the procedure. Physicians may request additional appointments for up to 3 more patients. Based on observational data the conditional probabilities for the appointment requests for these patients are approximately 0.8, 0.5, 0.3.

Table 4.2 includes the optimal allowances for each customer for different choices of the cost ratio, $\alpha = c^\ell / c^w$ and the optimal schedules are depicted in Figure 4.4. The optimal allowances for routine patients form a dome shape as in static scheduling problems since these patients are known to be scheduled with certainty. As $\alpha$ decreases from 10 to 1, the allowances between patient arrivals increase, and for $\alpha = 0.1$ allowances are nearly identical. In other words, as the cost of overtime increases, patient inter-arrival times decrease, and as the cost of waiting time increases patient inter-arrival times increase.

Figure 4.5 illustrates the results of an experiment done to observe the effects of the changes in number of add-on patients on the optimal schedule. Numbers in brackets in the legend denote routine patients and add-on patients respectively. For this experiment we used uniformly distributed service times. The appointment request probabilities, $q_i$, for add-on customers are selected to have decreasing values. Our results show that the optimal schedule is sensitive to the number of routine vs. add-on patients. As the expected number of patients in the system decreases from 12 routine and 0 add-on (12,0) to (3,9), the optimal time allowances for patients
Table 4.2: Optimal time allowances for varying cost ratios for 5 routine and 3 add-on patients with LogNormal service times.

<table>
<thead>
<tr>
<th>Allowances</th>
<th>$\alpha = 10$</th>
<th>$\alpha = 1$</th>
<th>$\alpha = 0.1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>20.96</td>
<td>25.99</td>
<td>31.93</td>
</tr>
<tr>
<td>$x_2$</td>
<td>23.98</td>
<td>27.95</td>
<td>32.67</td>
</tr>
<tr>
<td>$x_3$</td>
<td>24.65</td>
<td>27.73</td>
<td>32.57</td>
</tr>
<tr>
<td>$x_4$</td>
<td>24.53</td>
<td>27.59</td>
<td>32.71</td>
</tr>
<tr>
<td>$x_5$</td>
<td>23.91</td>
<td>26.42</td>
<td>32.54</td>
</tr>
<tr>
<td>$x_6$</td>
<td>21.93</td>
<td>26.56</td>
<td>32.29</td>
</tr>
<tr>
<td>$x_7$</td>
<td>23.12</td>
<td>26.26</td>
<td>32.13</td>
</tr>
</tbody>
</table>

Figure 4.4: Structure of the optimal solution of 5 routine and 3 add-on patients with different cost ratio $\alpha = c^f/c^w$, $Z_i \sim 3 + \text{Lognormal}(23.55, 11.89)$, $d = 150$.

early in the day increase and those for patients later in the day decrease (for most patients). Note that the time allowances in the presence of add-on patients are not monotonic, and in some cases do not exhibit the dome shape observed for the static appointment scheduling problems.

4.3.2 Computational Performance of Proposed Methods

We test the algorithms we propose on (D-ASP) since it is the more computationally challenging of the two models. We solve (D-ASP) with variants of the ND algorithm combined with our multi-cut approach, two-variable algorithm of Dyer [29], and the valid inequalities described in Section 4.2.2. We report the solution times and the number of iterations for three instances ($n_U = 10, 20, 30$). The appointment request probabilities, $q_i$, are assumed to be decreasing as the number of customers gets larger which is a natural attribute of a scheduling system with add-on customers. The daily session length, $d$, is chosen to be less than the product of mean service duration and $n_U$ so that expected overtime is nonzero (to represent the potentially
congested nature of appointment scheduling systems).

From Table 4.3, we conclude that the multi-cut version of the ND algorithm performs best in solution time and in total number of iterations of the ND algorithm for all problem instances. The two-variable algorithm also shows promising performance. Based on our experiments it provides similar results to CPLEX 11.0, indicating that perhaps CPLEX includes an implementation of Dyer’s algorithm, or a similar algorithm to take advantage of the structure of 2-variable LPs. In general, for large problems it takes more computational effort to solve lognormal data than uniform.

Table 4.4 shows the effects of the valid inequalities (VI-1), (VI-2) and (VI-3) added to the master problems in the ND algorithm for varying cost ratios, α. Adding valid inequalities (VI-1) and (VI-2) did not result in significant changes in total computation time. On the contrary, including more cuts in the master problems decreased the efficiency of the solution procedure for many of the test instances. Adding valid inequality set (VI-3) generally provided modest improvement in computational performance of ND.

Table 4.5 presents the results for similar numerical experiments with $d = 0$. By Proposition 1, this is consistent with the objective of minimizing a weighted sum of expected customer waiting time and expected server idle time. As can be seen from Table 4.5, the valid inequalities (VI-1) and (VI-2) provide greater benefit than observed in $d > 0$ experiments. (VI-1) performs better than (VI-2) except for three cases ($n_U = 10, \alpha = 1$; $n_U = 20, \alpha = 0.1$; $n_U = 30, \alpha = 1$). This may be due to two reasons. First, (VI-1) and (VI-2) relax the dependence of overtime on the time allowances in early stages when $d > 0$. Second, (VI-1) preserves the two variable structure, while at the same time involving fewer additional inequalities than (VI-2).
Table 4.3:  Computational performance of standard ND, multi-cut version of ND, and the two-variable algorithm implemented within ND.

<table>
<thead>
<tr>
<th></th>
<th>Number of Iterations</th>
<th>CPU Time (in seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( n_U = 10 )</td>
<td>( n_U = 20 )</td>
</tr>
<tr>
<td></td>
<td>(d=200)</td>
<td>(d=400)</td>
</tr>
<tr>
<td>ND</td>
<td>244</td>
<td>432</td>
</tr>
<tr>
<td>U(20, 40) ( \alpha = 10 )</td>
<td>186</td>
<td>244</td>
</tr>
<tr>
<td>Two-variable ND</td>
<td>254</td>
<td>406</td>
</tr>
<tr>
<td>ND</td>
<td>192</td>
<td>330</td>
</tr>
<tr>
<td>U(20, 40) ( \alpha = 1 )</td>
<td>106</td>
<td>184</td>
</tr>
<tr>
<td>Two-variable ND</td>
<td>186</td>
<td>290</td>
</tr>
<tr>
<td>ND</td>
<td>190</td>
<td>302</td>
</tr>
<tr>
<td>U(20, 40) ( \alpha = 0.1 )</td>
<td>96</td>
<td>176</td>
</tr>
<tr>
<td>Two-variable ND</td>
<td>186</td>
<td>290</td>
</tr>
<tr>
<td>ND</td>
<td>238</td>
<td>436</td>
</tr>
<tr>
<td>LogN(3.34, 0.325) ( \alpha = 10 )</td>
<td>182</td>
<td>320</td>
</tr>
<tr>
<td>Two-variable ND</td>
<td>254</td>
<td>466</td>
</tr>
<tr>
<td>ND</td>
<td>180</td>
<td>432</td>
</tr>
<tr>
<td>LogN(3.34, 0.325) ( \alpha = 1 )</td>
<td>120</td>
<td>230</td>
</tr>
<tr>
<td>Two-variable ND</td>
<td>202</td>
<td>370</td>
</tr>
<tr>
<td>ND</td>
<td>200</td>
<td>392</td>
</tr>
<tr>
<td>LogN(3.34, 0.325) ( \alpha = 0.1 )</td>
<td>112</td>
<td>216</td>
</tr>
<tr>
<td>Two-variable ND</td>
<td>188</td>
<td>362</td>
</tr>
</tbody>
</table>

Table 4.4:  Computational performance of Standard ND and Standard ND with (VI-1), (VI-2) and (VI-3) \((Z_i \sim U(20, 40))\).

<table>
<thead>
<tr>
<th></th>
<th>Number of Iterations</th>
<th>CPU Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( n_U = 10 )</td>
<td>( n_U = 20 )</td>
</tr>
<tr>
<td></td>
<td>(d=200)</td>
<td>(d=400)</td>
</tr>
<tr>
<td>ND</td>
<td>244</td>
<td>432</td>
</tr>
<tr>
<td>( \alpha = 10 ) ND with (VI-1)</td>
<td>224</td>
<td>456</td>
</tr>
<tr>
<td>ND with (VI-2)</td>
<td>264</td>
<td>460</td>
</tr>
<tr>
<td>ND with (VI-3)</td>
<td>232</td>
<td>370</td>
</tr>
<tr>
<td>ND</td>
<td>210</td>
<td>334</td>
</tr>
<tr>
<td>( \alpha = 1 ) ND with (VI-1)</td>
<td>228</td>
<td>344</td>
</tr>
<tr>
<td>ND with (VI-2)</td>
<td>210</td>
<td>410</td>
</tr>
<tr>
<td>ND with (VI-3)</td>
<td>188</td>
<td>306</td>
</tr>
<tr>
<td>ND</td>
<td>190</td>
<td>302</td>
</tr>
<tr>
<td>( \alpha = 0.1 ) ND with (VI-1)</td>
<td>180</td>
<td>304</td>
</tr>
<tr>
<td>ND with (VI-2)</td>
<td>170</td>
<td>344</td>
</tr>
<tr>
<td>ND with (VI-3)</td>
<td>174</td>
<td>284</td>
</tr>
</tbody>
</table>

4.3.3 Convergence of the Lower Bound

In this section we provide results of an experiment illustrating the convergence of ND (standard ND, and ND with added (VI-3)). Figure 4.6 depicts how quickly the bound improves with
Table 4.5: Computational performances of ND and ND with (VI-1), (VI-2) and (VI-3) ($d = 0, Z_i \sim U(20, 40)$).

<table>
<thead>
<tr>
<th></th>
<th>Number of Iterations</th>
<th>CPU Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$n_U = 10$ ($d=200)$</td>
<td>$n_U = 20$ ($d=400)$</td>
</tr>
<tr>
<td>ND</td>
<td>192</td>
<td>370</td>
</tr>
<tr>
<td>$\alpha = 10$ ND with (VI-1)</td>
<td>172</td>
<td>328</td>
</tr>
<tr>
<td>ND with (VI-2)</td>
<td>178</td>
<td>332</td>
</tr>
<tr>
<td>ND with (VI-3)</td>
<td>172</td>
<td>350</td>
</tr>
<tr>
<td>$\alpha = 1$ ND with (VI-1)</td>
<td>212</td>
<td>434</td>
</tr>
<tr>
<td>ND with (VI-2)</td>
<td>186</td>
<td>412</td>
</tr>
<tr>
<td>ND with (VI-3)</td>
<td>182</td>
<td>378</td>
</tr>
<tr>
<td>$\alpha = 0.1$ ND with (VI-1)</td>
<td>202</td>
<td>390</td>
</tr>
<tr>
<td>ND with (VI-2)</td>
<td>190</td>
<td>376</td>
</tr>
<tr>
<td>ND with (VI-3)</td>
<td>208</td>
<td>348</td>
</tr>
</tbody>
</table>

respect to the number of iterations, $\nu$. Although adding (VI-3) did not improve the solution performance significantly, it improved the bound on the optimal solution considerably at early iterations. With the addition of valid inequalities (VI-3), ND method reaches approximately 70% of the optimum lower bound within approximately 50 iterations as opposed to 70 iterations with standard ND. We note that this is encouraging for solving problems in which (D-ASP) is a sub-problem and efficient computation of lower bounds is important.

Figure 4.6: Comparison of convergence for ND, and ND with added (VI-3) ($n_U = 40, Z_i \sim U[20, 40], d = 450, q_i = 0.5$).
4.3.4 Value of the Stochastic Solution

We compute the value of the stochastic solution (VSS) for (D-ASP) for several test instances. Table 4.6 shows the relative benefit of solving the stochastic problem for a 30 customer problem with varying numbers of routine and add-on customers, and with varying cost ratios, $\alpha$. Note that the VSS is typically increasing in $\alpha$ meaning that solving the stochastic problem is increasingly important as the customer waiting cost increases with respect to overtime cost.

Table 4.6: Value of the Stochastic Solution for several test instances with $Z_i \sim U[20, 40]$ and $q_i = 0.5$ for add-on requests.

<table>
<thead>
<tr>
<th>Number of Patients (Routine, Add-on)</th>
<th>VSS (%)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$d = 0$</td>
<td>$d = 200$</td>
</tr>
<tr>
<td>$(0,30)$</td>
<td>$\alpha = 10$</td>
<td>$\alpha = 0.1$</td>
</tr>
<tr>
<td></td>
<td>$\alpha = 1$</td>
<td>$\alpha = 0.1$</td>
</tr>
<tr>
<td>$(10,30)$</td>
<td>$0.61$</td>
<td>$10.65$</td>
</tr>
<tr>
<td></td>
<td>$66.95$</td>
<td>$1.40$</td>
</tr>
<tr>
<td></td>
<td>$23.63$</td>
<td>$79.41$</td>
</tr>
<tr>
<td>$(20,30)$</td>
<td>$0.38$</td>
<td>$18.45$</td>
</tr>
<tr>
<td></td>
<td>$75.46$</td>
<td>$0.50$</td>
</tr>
<tr>
<td></td>
<td>$23.63$</td>
<td>$80.33$</td>
</tr>
</tbody>
</table>

4.4 Conclusions

In this chapter we proposed a new stochastic programming model for the dynamic appointment scheduling problem. The objective was to find the optimal appointment times when the number of customers are not known since a number additional customers are allowed to request appointments after an initial schedule has been created.

We proposed several methods based on ND that take advantage of the underlying structure of the problem including additional valid inequalities, a fast method for solving 2-variable LPs, and an adaptation of multi-cut outer linearization. We conducted a series of numerical experiments with varying cost ratios, service time distribution types, and appointment request probabilities. Computational improvements were observed, and they are particularly encouraging for future applications that involve solving problems in which (D-ASP) is a subproblem, and efficient computation of lower bounds is important (e.g. a branch-and-bound implementation for problems that consider assignment of customers among multiple servers or multiple days). Our results indicate that the multi-cut implementation of ND for (D-ASP) also gives significant computational advantage.

Our numerical experiments show that the dome shape observed in the static scheduling case is preserved for routine customer appointments. However, as the relative number of routine customers decreases and add-on customers increases, the inter-arrival times increase for the
routine patients scheduled to arrive early in the day, and decrease for the add-on patients scheduled to arrive later in the day. In other words, the allowances for the customers that request appointments with certainty increases whereas the allowances for the customers that request appointments probabilistically decreases. In this case, the minimum total cost can be achieved by having less waiting time for the customers early in the schedule, and less overtime by considering the potential for not requesting appointments for the customers later in the schedule. Finally, we observe very high VSS for some problem instances, indicating that solution of the stochastic program is important in the dynamic scheduling context.
Chapter 5

Dynamic Appointment Sequencing and Scheduling

In Chapter 4, (D-ASP) focused on the dynamic scheduling of customers that arrive stochastically to request appointments to a single server with uncertain service durations. Customers were scheduled sequentially as they requested appointments in first-come-first-serve (FCFS) order. In this chapter, we relax the assumption that the sequence is FCFS. In other words we assume that the customers are not necessarily scheduled in the order of their appointment requests. Thus, in this chapter, the appointment request sequence may differ from the appointment arrival sequence.

Figure 5.1 illustrates the evolution of an online appointment schedule over time for a case in which up to 5 customers are scheduled. Figure 5.1(A) illustrates the special case of a first-come-first-serve (FCFS) policy which represents the dynamic appointment scheduling problem (D-ASP) investigated in Chapter 4. Figure 5.1(B) illustrates the more general case, which we consider in this chapter, in which the sequence is not fixed a priori. Note that customers are scheduled in order of their appointment requests but their appointment times do not necessarily follow that order. We formulate a two-stage stochastic mixed integer programming model (MIP) for this general on-line appointment scheduling problem.

We investigate this problem because there are a number of health service environments in which FCFS arrival scheduling is not necessarily optimal. Given the dynamic nature of the scheduling environment, in the presence of different priority patients, for example, the scheduler may consider the relative importance of the patients when reserving the appointment times. Since scheduling is done sequentially and rescheduling is uncommon in most service systems, sequencing decisions are an important part of appointment scheduling decisions. While each scheduling decision is being made, the future uncertain arrivals of patients, perhaps with varying priority, must be considered.
In the static (off-line) context, scheduling appointments with different patient classes has received recent attention from several researchers. Previous studies have considered several patient classifications according to characteristics such as new/returning patients, child/adult patients, or according to service durations (e.g. high vs. low variance) and/or procedure types. In the context of surgery scheduling, for example, surgeries often are classified in two categories: elective and urgent. For elective cases, surgery may be planned well in advance (e.g. months) to be performed on a future date. For non-elective cases, on the other hand, the surgery is unanticipated (either an urgent add-on case or an emergent case). These cases must be worked into the existing schedule, either by using intentionally reserved or otherwise available space in the schedule, or by creating space by canceling previously scheduled elective cases. In some health care environments threshold policies are applied. According to these policies,
lower priority patients (i.e. outpatients) are scheduled until a threshold is reached. Remaining capacity is reserved for higher priority patients that may arrive in the future (i.e. emergency, inpatient) [60]. Some hospitals allocate separate ORs for emergencies and add-ons, whereas others allow slack time in their schedule [34, 85, 49].

Cayirli and Veral [16] developed a simulation model to determine the sequence and schedule for the new and returning patients in an ambulatory care system. They tested several sequencing rules including FCFS, alternating between new and returning patients, sequencing new patients at the beginning and sequencing returning patients at the beginning etc. In addition to these sequencing rules, several scheduling rules to determine the appointment allowances are also tested. These rules include allocating equal intervals between patients, double-booking the first two patients (Bailey’s Rule), and scheduling two patients at a time with equal intervals. They concluded that sequencing decisions have more impact on the performance of the system than the appointment scheduling rules. Later, in another study, Cayirli et al. considered different environmental characteristics such as no-show rates, the ratio of new patients to returning patients and walk-ins [17]. They concluded that FCFS is not necessarily optimal when there are different patient classes. They found that different sequencing and scheduling rules should be selected depending on the environmental characteristics. However, in general, scheduling two patients at the beginning of the day and sequencing new patients before returning ones result in lowest cost at all environments.

More recently, Zonderland et al. [95] studied the tradeoff between cancelation of scheduled elective surgeries to accommodate semi-urgent arrivals and the unused OR time that is reserved for uncertain semi-urgent surgeries. The authors used an infinite horizon Markov decision process to determine the number of slots to be reserved for semi-urgent arrivals. They found that when the cost of canceling elective surgeries is higher than the cost of OR idle time, the optimal policy is to reserve appointment slots for a number of semi-urgent arrivals in advance but postpone the remaining semi-urgent surgeries. When the cost of OR idle time is higher, the optimal policy is to cancel elective surgeries to accommodate semi-urgent surgeries.

The above referenced studies provide empirical evidence that sequencing decisions are an important consideration when patient characteristics are not identical. In the next section we consider a special case for which FCFS can be shown to be optimal. We also provide a counter example that demonstrates that FCFS is not optimal in all cases. To prove this, we compare FCFS to a scheduling rule referred to as add-ons-first-scheduling (AOFS). Next, we present a model formulation for a small example of a multistage stochastic program that is based on AOFS. These results motivate our research on the general on-line appointment scheduling problem where the sequence is not fixed a priori. We formulate this general problem as a two-stage stochastic MIP with binary decision variables representing patient sequencing decisions. We show how the multistage decision process of on-line scheduling can be expressed
as a two-stage recourse problem. Next, we discuss an implementation of the integer L-shaped method and several structural properties of the model that can be exploited for computational efficiency. Finally, we present results that demonstrate the optimal sequencing and scheduling decisions and computational performance of the solution methods.

5.1 Motivation for Dynamic Appointment Sequencing and Scheduling

In this section, we present special cases for which FCFS is optimal and suboptimal. We start by providing a proof for the case of independent and identically distributed (i.i.d.) service durations. We consider a simple case where \( n = 2, \ d = 0, \ Z_1 \) and \( Z_2 \) are i.i.d. and customer 1 requests appointment with probability 1 and customer 2 requests an appointment with probability \( q \). We begin by defining the following two problems:

**FCFS:** The first customer in the appointment request sequence is scheduled first. A second customer requests appointment with probability \( q \) and this customer is scheduled to arrive after the first customer. Therefore \( a_1 = 0, \ w_{1j}(\omega) = 0 \ \forall j = 1, 2 \). The decision variable \( x_1 \) denotes the time allowance between the first and the second customer. The minimization problem is as follows:

\[
z_1(x_1) = \min_{x_1} \{ (1 - q)(c_w E[w_{1,1}(\omega)] + c^\ell E[\ell_1(\omega)]) + q(c_w E[w_{1,2}(\omega) + w_{2,2}(\omega)] + c^\ell E[\ell_2(\omega)]) \\
= \min_{x_1} \{ (1 - q)c^\ell E[(Z(\omega) - d)^+] + q(c_w E[(Z(\omega) - x_1)^+] + c^\ell E[(w_2(\omega) + Z(\omega) - x_1)^+)]
+ Z(\omega) + x_1 - d)^+)) \}
= \min_{x_1} \{ (1 - q)c^\ell E[Z(\omega)] + q(c_w E[(Z(\omega) - x_1)^+] + c^\ell E[(w_2(\omega) + Z(\omega) + x_1)^+])
= \min_{x_1} \{ (1 - q)c^\ell E[Z(\omega)] + q(c_w E[(Z(\omega) - x_1)^+] + c^\ell E[(Z(\omega) - x_1)^+
+ Z(\omega) + x_1)^+]) \}. \quad (5.1)
\]

**AOFS:** The second customer in the an appointment request sequence requests appointment with probability \( q \) after the first customer requests appointment. However, the second customer is scheduled to arrive first. Therefore \( a_2 = 0, \ w_2 = 0, \ \ell_1 = (Z(\omega) + x_2 - d)^+ \). The minimization
problem is as follows:

\[ z_2(x_2) = \min_{x_2} \{(1-q)\left(c w E[w_{2,1}(\omega)] + c^\ell E[\ell_1(\omega)]\right) + q\left(c w E[w_{2,2}(\omega) + w_{1,2}(\omega)] + c^\ell E[\ell_2(\omega)]\right) \]

\[ = \min_{x_2} \{(1-q)\left(c^\ell E[(Z(\omega) + x_2 - d)^+] + q\left(c w E[(Z(\omega) - x_2)^+] \right) + c^\ell E[(w_1(\omega) + Z(\omega) + x_2 - d)^+]\right) \}

\[ = \min_{x_2} \{(1-q)\left(c^\ell E[(Z(\omega) + x_2)^+] + q\left(c w E[(Z(\omega) - x_2)^+] \right) + c^\ell E[(Z(\omega) - x_2)^+] \}

\[ + c^\ell E[(Z(\omega) - x_2)^+] + Z(\omega) + x_2)\] \}\}. \quad (5.2)

Based on the above problem definitions, (5.1) and (5.2), we prove the following proposition.

**Proposition 7** For \( n = 2 \) with i.i.d. service durations, if the second customer requests an appointment with probability \( q \), FCFS is better than AOFS.

**Proof:** Let the optimal solutions for FCFS and AOFS be \( x_1^* \) and \( x_2^* \) and the optimal objective function values be \( z_1(x_1^*) \) and \( z_2(x_2^*) \), respectively. By convexity of the expectation of waiting and overtime, it follows that

\[ z_1(x_1^*) \leq z_1(x_2^*) \]

\[ = (1-q)\left(c^\ell E[Z(\omega)] + q\left(c w E[(Z(\omega) - x_2^* + Z(\omega) + x_2^*)^+] \right) \]

\[ \leq (1-q)\left(c^\ell E[(Z(\omega) + x_2^*)^+] + q\left(c w E[(Z(\omega) - x_2^*)^+] + c^\ell E[(Z(\omega) - x_2^*)^+] + Z(\omega) + x_2^*)^+] \right) \]

\[ = z_2(x_2^*). \quad \square \quad (5.3) \]

This shows that if the two customers are identical in their service distributions and their waiting time cost coefficients, the appointment request sequence should be preserved for scheduling the arrivals, i.e. customers should be scheduled in FCFS order. We conjecture that this rule holds for problem sizes \( n > 2 \) with the same problem characteristics. In Section 5.4 we provide the results of numerical experiments that support this conjecture.

### 5.1.1 Dynamic Appointment Scheduling Model with AOFS

In this section we extend the (D-ASP) model from the FCFS to the AOFS queue discipline. We illustrate the problem with an example with two routine customers and one high priority customer that is to be scheduled first if they request an appointment. Customers are indexed by their appointment requests. The sequencing and scheduling process is illustrated in Figure 5.2 and is defined by the following process:

- The first customer requests an appointment and is scheduled to arrive at time \( a_1 \)
• The second customer requests an appointment with probability $q_2$ and is scheduled to arrive at $a_2$, $a_1 < a_2$

• The third customer requests an appointment with probability $q_3$ and is scheduled to arrive at time $a_3$ before customer 1 and customer 2 ($a_3 < a_1 < a_2$)

Figure 5.2: Illustration of the sequencing and scheduling problem with probabilistic arrival of 3 customers given AOFS sequence.

The scenarios for the decision process are represented by the tree in Figure 5.3. According to the AOFS sequence, the third customer is scheduled to arrive (if they call for an appointment) before the first and the second customers.

We adapt the stochastic programming formulation of Chapter 4 for the appointment scheduling problem with the AOFS queue discipline. We assume that $n^R$ routine customers request appointments with probability 1. Then each of $n^U - n^R$ add-on customers request an appointment sequentially with probability $q_i$, $i = n^R, n^R + 1, \ldots, n^U$. The appointment request sequence within the class of routine customers and the class of urgent customers are each preserved in the appointment arrival sequence. However, add-on customers are scheduled to arrive before routine customers. In other words, the priority sequence of the customers is assumed to be known before scheduling, thus the sequence is determined a priori and will be fixed before optimizing the schedule.

Similar to (D-ASP), of Chapter 4, this problem can be formulated as an M-SLP. However, instead of time allowances, $x_i$, between customer arrivals, the formulation is based on decision variables that are appointment times, $a_i$. Note that appointment times for each customer can be determined by adding the appointment time of the previous customer and that customer’s
time allowance, i.e. \( a_i = a_{i-1} + x_{i-1}, \forall i = 2, \ldots, n_U \). The M-SLP formulation of this problem is as follows:

\[
\min_{a_1, a_2, \ldots, a_{n_R}} \{ (1-q_{n_R+1})Q_{n_R}(a_1, a_2, \ldots, a_{n_R}) + \min_{a_{n_R+1}} \{ (q_{n_R+1})(1-q_{n_R+2})Q_{n_R+1}(a_{n_R+1}) \\
+ \ldots + \min_{a_{n_R+U}} \{ \prod_{j=n_R+1}^{n_U} (q_j)Q_{n_U}(a_{n_U}) \} \} \ldots \}.
\]

The terminal subproblems at stage \( j = n_R \) can be formulated as the subproblems given in (4.5), in Chapter 3, assuming FCFS order is preserved within the routine customers. For the following stages with add-on customers, the terminal subproblems at stage \( j \), for \( j \geq n_R + 1, \ldots, n_U \), can be formulated as follows:

---

**Figure 5.3:** Scenario tree for M-SLP problem with 3 customers given that customer 3 one has higher priority.
\[ Q_j(a_j, \omega_j) = \min_{w,s,l} \left\{ c^w \sum_{i=1}^{\omega_j} w_{j,i}(\omega_j) + c^l \ell_j(\omega_j) \right\} \quad (5.4) \]

\[ \text{s.t} \quad w_{j,n^{R+1}}(\omega_j) = 0 \quad (5.5) \]

\[ w_{j,n^{R+2}}(\omega_j) \geq Z_{n^{R+2}}(\omega_j) - a_{n^{R+2}} + a_{n^{R+1}} \quad (5.6) \]

\[ - w_{j,n^{R+2}}(\omega_j) + w_{j,n^{R+3}}(\omega_j) \geq Z_{n^{R+3}}(\omega_j) - a_{n^{R+3}} + a_{n^{R+2}} \quad (5.7) \]

\[ \vdots \]

\[ - w_{j,j-1}(\omega_j) + w_{j,j}(\omega_j) \geq Z_{j-1}(\omega_j) - a_j + a_{j-1} \quad (5.8) \]

\[ - w_{j,j}(\omega_j) + w_{j,1}(\omega_j) \geq Z_j(\omega_j) - a_1 + a_j \quad (5.9) \]

\[ - w_{j,1}(\omega_j) + w_{j,2}(\omega_j) \geq Z_1(\omega_j) - a_2 + a_1 \quad (5.10) \]

\[ \vdots \]

\[ - w_{j,n^{R-1}}(\omega_j) + w_{j,n^{R}}(\omega_j) \geq Z_{n^{R-1}}(\omega_j) - a_n^{R} + a_{R-1} \quad (5.11) \]

\[ - w_{j,n^{R}}(\omega_j) + \ell_j(\omega_j) \geq Z_{n^{R}}(\omega_j) + a_n^{R} - d \quad (5.12) \]

\[ w_{j,i}(\omega_j) \geq 0 \forall i, \ell_j(\omega_j) \geq 0. \]

where \( a_{n^{R+1}} = 0 \) since the first add-on customer is scheduled to arrive at the beginning of the day in the AOFS sequence. The corresponding dual variables of constraints \((5.5), (5.6), \ldots, (5.12)\) are \( \pi_{j,n^{R+1}}, \pi_{j,n^{R+2}}, \ldots, \) and \( \pi_{j,j+1} \) respectively. The dual solution to terminal subproblems can be computed by the following backward recursion:

\[ \pi_{j,i}(a_j, \omega_j) = \begin{cases} 
0 & w_{j,i+1}(\omega_j) = 0, \\
c^w_{i+1} + \pi_{j,i+1}(a_j, \omega_j) & w_{j,i+1}(\omega_j) > 0,
\end{cases} \quad (5.13) \]

for \( i = 1, \ldots, n^{R-1} \) and \( i = n^{R+1}, \ldots, j - 1 \),

\[ \pi_{j,n^{R}}(a_j, \omega_j) = \begin{cases} 
0 & w_{j,n^{R}}(\omega_j) = 0, \\
c^w_{n^{R}} + \pi_{j,j+1}(a_j, \omega_j) & w_{j,n^{R}}(\omega_j) > 0,
\end{cases} \quad (5.14) \]

\[ \pi_{j,j}(a_j, \omega_j) = \begin{cases} 
0 & w_{j,j}(\omega_j) = 0, \\
c^w_{j} + \pi_{j,1}(a_j, \omega_j) & w_{j,j}(\omega_j) > 0,
\end{cases} \quad (5.15) \]
\[
\pi_{j,j+1}(a_{nR}, a_{nR+1}, \omega_j) = \begin{cases} 
0 & \ell_j(\omega_j) = 0, \\
c^\ell & \ell_j(\omega_j) > 0.
\end{cases}
\] (5.16)

The closed form expressions for the dual solutions of the subproblems offer considerable computational savings in solving the M-SLP using nested decomposition.

### 5.1.2 Counter-Example: FCFS vs. AOFS

In this section, we present some preliminary numerical results for a larger problem (greater than 2 customers) based on an instance for which FCFS is suboptimal. For this instance we consider a scheduling problem with \(n^R = 9\) routine and 3 add-on customers \(n^U = 12\). Routine customers request appointments deterministically in the first stage, add-on customers may or may not request an appointment. In this example the conditional probabilities of each add-on customer requesting an appointment are 0.8, 0.7, and 0.6 respectively. Again, we assume that add-on customers request appointments after routine customers, due to the nature of their urgent (short notice) conditions; however, if the add-on customers request an appointment, they will be scheduled and receive service before the routine customers. This higher priority for scheduling add-on customers is reflected in their waiting time costs which are \(c_{10}^w = c_{11}^w = c_{12}^w = 1000\), and \(c_1^w = c_2^w = \ldots = c_9^w = 1\).

Table 5.1 compares the numerical results for both FCFS and AOFS. The results indicate that AOFS may produce better (less costly) schedules than FCFS when waiting times for add-on customers are significantly higher than routine customers. As can be seen from the optimal appointment times, the first routine customer is scheduled at the beginning of the day \((a_1 = 0)\) in the FCFS case, whereas in the AOFS case, this customer is scheduled to arrive at time \(a_1 = 39.4\). In the AOFS schedule, on the other hand, priority is given to add-on customers, 10, 11 and 12, and therefore the first add-on customer is scheduled at the beginning of the day \((a_{10} = 0)\). Another important observation from Table 5.1 is that for AOFS some of the routine customers may be scheduled before some add-on customers as in the case for \((a_1, a_2\) and \(a_{11}, a_{12}\)). This means that these routine customers \((i=1\) and \(i=2\)) may receive service at their appointed times if add-on customers \((i=11\) and \(i=12\)) do not request appointments.

The example presented here represents a scheduling environment in which some high priority customers are served first even though there are some other customers scheduled to arrive before the high priority customers. In such cases, the high priority customers (with high waiting time costs) are served first at the expense of making previously scheduled customers wait. This is a direct consequence of the fact that appointment decisions must be made sequentially with imperfect information about the future. Figure 5.4 depicts the optimal schedule for the same
Table 5.1: Optimal Appointment Schedules with FCFS and AOFS for a 9 routine + 3 add-on customers problem with $Z_i \sim 3 + \text{Lognormal}(23.55, 11.89)$.

<table>
<thead>
<tr>
<th>Appointment times</th>
<th>FCFS</th>
<th>AOFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_1$</td>
<td>0.0</td>
<td>39.38</td>
</tr>
<tr>
<td>$a_2$</td>
<td>22.02</td>
<td>67.14</td>
</tr>
<tr>
<td>$a_3$</td>
<td>47.37</td>
<td>93.62</td>
</tr>
<tr>
<td>$a_4$</td>
<td>73.65</td>
<td>119.49</td>
</tr>
<tr>
<td>$a_5$</td>
<td>100.30</td>
<td>145.05</td>
</tr>
<tr>
<td>$a_6$</td>
<td>126.94</td>
<td>170.13</td>
</tr>
<tr>
<td>$a_7$</td>
<td>153.52</td>
<td>194.85</td>
</tr>
<tr>
<td>$a_9$</td>
<td>182.49</td>
<td>210.01</td>
</tr>
<tr>
<td>$a_9$</td>
<td>213.76</td>
<td>242.06</td>
</tr>
<tr>
<td>$a_{10}$</td>
<td>261.94</td>
<td>0.0</td>
</tr>
<tr>
<td>$a_{11}$</td>
<td>301.18</td>
<td>37.33</td>
</tr>
<tr>
<td>$a_{12}$</td>
<td>340.09</td>
<td>74.49</td>
</tr>
<tr>
<td>$Z^*$</td>
<td>748.7</td>
<td>687.9</td>
</tr>
</tbody>
</table>

problem presented above with varying waiting costs for routine and add-on customer. Note that as the cost of waiting for add-on customers increases, add-on customers are scheduled further apart to minimize their waiting time at the expense of a large amount of waiting time that may occur for the routine customers. Crossing of lines in Figure 5.4 indicate a change in the sequence of customer arrivals.

5.2 Dynamic Appointment Sequencing and Scheduling Model

In this section we provide a general dynamic appointment scheduling problem for an online appointment scheduling system in which the number of customers to be scheduled is not known in advance. As opposed to our previous dynamic scheduling models with fixed sequences (FCFS and AOFS), we present a model in which the sequence is not fixed a priori. A maximum of $n$ customers can be scheduled, where $n$ is an upper bound on the total number of customers that could be served on a particular day. The appointment scheduling process is as follows: Customers are quoted appointment times, as appointment requests arise, up to a maximum of $n$ appointments for a particular day. The sequence of appointments may change over time as the appointment schedule evolves; however, once an appointment time is quoted for a given customer it cannot be change.

In our models of Chapter 3 and Chapter 4, the waiting time is the amount of time customers wait between their scheduled arrival times and the start of their service. This waiting time has been previously defined as the direct waiting time [39]. It is the amount of time a customer waits on-site relative to their arrival time at the stochastic server. However, in some environments,
indirect waiting time, the total amount of time between the appointment request and the scheduled arrival time is more important than direct waiting time [39] [56]. For instance, the amount of indirect waiting time must be minimized if the medical condition of an add-on patient is highly critical (e.g. a trauma case that arrives at a surgical suite). In such cases, there is a cost associated with not scheduling add-on patients to arrive early in the day.

To include indirect waiting cost in our new model, we include a new vector of decision variables, $a$, which defines the appointment times for each customer. The allowance between two consecutive customers in the sequence will be the difference between the arrival times of these customers. To include the indirect waiting cost, we define a new vector of cost coefficients, $c^a$, associated with the new appointment time variables in the first stage. (Note that, in the previous chapters’ models, the objective had no cost associated with the first stage decisions).

We use the following notation, where upper case indicates random variables and bold face is used to denote vectors:

**Model Parameters:**

$n$: number of customers to be scheduled

$\omega$: index for service duration scenarios

$p_j$: probability of having $j$ customers in the system

Figure 5.4: Appointment times with varying waiting cost ratios for routine and add-on customers.
$Z(\omega)$: vector of random service durations for $n$ customers

d: session length to complete all customers before overtime is occurred

c$^w$: vector of waiting time cost coefficients for $n$ customers

c$^a$: vector of appointment time cost coefficients for $n$ customers

c$^\ell$: cost coefficient for overtime

c$^s$: cost coefficient for idle time

**Decision variables:**

$o_{j,i,i'}$: binary sequencing variable where $o_{i,i'} = 1$ if patient $i$ immediately precedes $i'$ at stage $j$, and $o_{i,i'} = 0$ otherwise

$x_{j,i,i'}$: time allowance for patient $i$ given that $i$ immediately precedes $i'$ (appointment inter-arrival time for patient $i$ and $i'$) at stage $j$

$a_{j,i,i'}$: arrival time of patient $i'$, given that $i$ immediately precedes $i'$ at stage $j$

$w_{j,i,i'}(\omega)$: waiting time of patient $i'$ given that patient $i$ immediately precedes $i'$ at stage $j$ under duration scenario $\omega$

$s_{j,i,i'}(\omega)$: server idle time between patient $i$ and $i'$, given that $i$ immediately precedes $i'$ at stage $j$ under duration scenario $\omega$

$\ell_j(\omega)$: overtime at stage $j$ with respect to session length $d$ under duration scenario $\omega$

Note that the vectors of decision variables $o, x, a, w, s$ are sequence dependent at each stage. Furthermore, the sequence may change from one stage to the next. This is due to the fact that in a given stage, the new customer might be scheduled between two customers that were sequenced consecutively. Thus, as the new customer is included in the sequence, one of the immediate precedence relationships in the sequence might be broken if this new customer is scheduled in between two previously sequenced customers. Since $x, a, w, s$ all depend on the particular sequence of customers they too depend on the stage of the decision process. Also note that, the probability of having $j$ customers in the system, $p_j$, is different than the conditional probability of appointment request for each customer, $q_i$, introduced in Chapter 3. The probability of having $j$ customers can be calculated by using probability of appointment requests, $q_i$, as follows:

$$p_j = (1 - q_j) \prod_{i=1}^{j} q_i.$$
In our model formulation, for each stage, \( j \), we include two dummy customers, customer 0 and \( j + 1 \). Customer 0 is always at the beginning of the sequence, and customer \( j + 1 \) is always at the end of the sequence. All other customers which are the original patients in the problem are sequenced between these two dummy patients at each stage. Introducing dummy customers ensures each customer, except dummy ones, is preceded and followed by another one regardless of the sequence. A valid sequence of appointments at any stage \( j \) is one that begins with the dummy customer 0 and ends with the dummy customer \( j + 1 \). Every customer must appear in the sequence exactly once. Between successive stages, the sequence of customers should not change, except the \( j^{th} \) customer will be inserted between two customers in the previous stage’s sequence or appear just before dummy customer \( j + 1 \).

The problem described above is by nature a multi-stage stochastic program, with the customer appointment requests defining the stages. However, multi-stage stochastic integer programs are widely regarded as computationally intractable. The presence of binary sequencing variables further complicates the solution of the problem. Instead we formulate our model as a two-stage stochastic program (2-SLP) in which binary (sequencing) decisions are restricted to the first stage. We use a novel set of constraints to enforce non-anticipativity of the appointment sequencing decisions across stages. This formulation has the benefit of a continuous and convex recourse function in the second stage, which allows for the application of the integer L-shaped method.

The 2-SLP formulation of the on-line appointment sequencing and scheduling problem is as follows:
min \sum_{j=1}^{n} p_{j} \left[ \sum_{i=1}^{j} \sum_{i'=1}^{j} c_{i,i'} a_{j,i,i'} \right] + Q(o, x, a) \tag{5.17}

s.t. \tag{D-ASSP) 

(5.18)

\sum_{i'=1}^{j} o_{j,0,i'} = 1 \quad \forall j \tag{5.19}

\sum_{i'=1}^{j+1} o_{j,i',j+1} = 1 \quad \forall j \tag{5.20}

\sum_{i'=1}^{j+1} o_{j,i',i'} = 1 \quad \forall j, i = 1, 2, \ldots, j \tag{5.21}

\sum_{i'=0}^{j+1} o_{j,i',i} = 1 \quad \forall j, i = 1, 2, \ldots, j \tag{5.22}

\sum_{i=0}^{j+1} \sum_{i'=0}^{j+1} o_{j,i,i'} = j + 1 \quad \forall j \tag{5.23}

o_{j,i,i'} - o_{j-1,i,i'} \leq 0 \quad \forall j, \forall i, i' < j \tag{5.24}

o_{j,i,i'} + o_{j,i',i} \leq 1 \quad \forall j, \forall i, i' < j \tag{5.25}

o_{j-1,i,i'} + o_{j,i',k} + o_{j,k,i} \leq 2 \quad \forall j, \forall i, i' < j, \forall k \leq j \tag{5.26}

o_{j,i,j} + o_{j,j,i,i'} - o_{j-1,i,i'} \leq 1 \quad \forall j, \forall i, i' < j \tag{5.27}

x_{j,i,i'} \leq M_{1} o_{j,i,i'} \quad \forall j, i, i' \tag{5.28}

a_{j,i,i'} \leq M_{1} o_{j,i,i'} \quad \forall j, i, i' \tag{5.29}

\sum_{i'=1}^{j+1} x_{j,i,i'} = \sum_{i'=1}^{j+1} a_{j,i,i'} - \sum_{i'=1}^{j+1} a_{j,i',i} \quad \forall j, i \tag{5.30}

\sum_{i'=1}^{j+1} a_{j,i,i'} = \sum_{i'=1}^{j} a_{j-1,i',i} \quad \forall j, i \tag{5.31}

x_{j,i,i'}, a_{j,i,i'} \geq 0, \quad o_{j,i,i'} \in \{0, 1\} \quad \forall j, i, i' \tag{5.32}

where

\[ Q(o, x, a) = E_{\xi(\omega)}[Q(o, x, a, \omega)], \tag{5.33} \]

and \( Q(o, x, a, \omega) \) defines the second stage scenario subproblem:
We refer to this problem as the \textit{dynamic appointment sequencing and scheduling} problem (D-ASSP). The vectors of allowances, \( x \), appointment times, \( a \), and binary sequencing variables \( o \), are all first stage decisions. The random job service time durations vector, \( Z(\omega) \), with support \( \Xi \in \mathbb{R}^n \), depends on outcomes indexed by \( \omega \in \Omega \). Customer waiting time, \( w(\omega) \in \mathbb{R}^{n^3} \), server idle time \( s(\omega) \in \mathbb{R}^{n^3} \) and overtime \( \ell(\omega) \in \mathbb{R}^n \) denote the second stage (recourse) decisions made after the observation \( \omega \) of random service durations. As noted above, \( x \), \( a \), and \( o \) depend on the appointment request scenario. Thus, treating them as first stage decisions implies they are made with perfect information about the number of future appointment arrivals. To correct this we use the concept of non-anticipativity constraints \cite{8}. The basic idea of such constraints is to require that decisions are the same for any history of the appointment request process.

The first stage constraints in the above formulation define feasible appointment schedules with respect to sequencing decisions. In the above formulation, constraint set (5.19) and ensures that dummy patient 0 is always at the beginning of the sequence at each stage and no other patient can be sequenced before dummy customer 0. Constraint set (5.20) ensures that dummy customer \( j + 1 \) is always at the end of the sequence at each stage and no other customer can be sequenced after dummy patient \( j + 1 \). Constraint sets (5.21) and (5.22) imply that each (non-dummy) customer is part of a feasible sequence, i.e. each patient comes before another and followed by another within a given stage. Constraint set (5.23) ensures that \( j + 1 \) precedence relationships exist at each stage \( j \) including the precedence relationships with dummy patients.

Even though the problem is formulated as a 2-SLP, the fact that its multistage nature is preserved in the first stage problem requires special constraints to ensure non-anticipativity of
the decisions across the stages. The constraint sets (5.24)-(5.27) ensure that the each stage’s sequencing decisions are made only based on the information available at that stage, i.e. the sequencing decisions made in the earlier stages. Constraint set (5.24) implies that if customer i directly precedes i’ at stage j − 1, this precedence is preserved unless if an insertion is made between i and i’ at stage j. Also, if there is not a direct precedence between i and i’ at stage j − 1, there cannot be one at stage j. Constraint set (5.25) prevents direct precedence violations in the same stage. If i comes before i’ at stage j, i’ cannot come before i. Constraint set (5.26) prevents the precedence violations in the next stage. This constraint eliminates the possible precedence violation but not the immediate precedence violations since those are taken care of in the previous constraints. Constraint set (5.27) ensures that the new customer (customer j) at stage j can be placed between i and i’ which had an immediate precedence relationship at stage j − 1. If i and i’ don’t have immediate precedence relationship at stage j − 1, customer j cannot have immediate precedence relationships with both i and i’ at stage j.

Constraints (5.28) and (5.29) ensure that corresponding allowances (x_{j,i,i’}) and appointment times (a_{j,i,i’}) will exist only if the customer i precedes i’ at stage j. M_{1} is sufficiently large to upper bound the optimal values of decisions x_{j,i,i’} and a_{j,i,i’}. Constraint (5.30) implies that the allowance for each customer is equal to the time difference between the appointment time of that customer and the appointment time of the following customer in the sequence. Constraint (5.31) enforces the appointment time for a customer to be preserved in the future stages. In other words, (5.31) ensure that the arrival time of patient i remains the same at each stage even though the sequence changes.

In the second stage, constraint sets (5.46) and (5.47) ensure that corresponding allowance, waiting and idling times will occur only if customer i directly precedes customer i’ at stage j. Constraint sets (5.48) and (5.49) determine the sequence dependent waiting times and the overtime at each stage j.

5.3 Structural Properties and Solution Methodology

The formulation expressed in (5.17) to (5.40) is a two-stage stochastic mixed integer program, with binary decisions in the first stage and a continuous second stage (linear program). The model has complete recourse since the recourse problem, Q(o, x, a, w) contains a positive linear basis, i.e., there is a feasible solution for any choice of o, x, a. In this section we present several properties of the dynamic appointment sequencing and scheduling model that we introduced in the previous section.
5.3.1 Alternative Formulation

The first stage problem given in the previous section is a mixed integer program which is difficult to solve. We present an alternative formulation for the first stage problem. The following single constraint can ensure that the direct precedence relationships are preserved with respect to the appointment scheduling decisions:

\[ o_{j,i,j} + o_{j,j,i'} \geq 2(o_{j-1,i,i'} - o_{j,i,i'}) \quad \forall j, \forall i, i' < j \]  
(5.41)

This new constraint replaces constraint sets (5.24)-(5.27), to obtain a feasible first stage solution. When both of the variables in the right hand side of (5.41) equal to 1, it indicates that the precedence relationship existing between customers \( i \) and \( i' \) at stage \( j - 1 \) is preserved at stage \( j \). In that case, none of the variables on the left hand side can take positive values due to constraints (5.19)-(5.22) which ensure that each customer can only be preceded and followed by only one customer. The left hand side of the constraint (5.41) represents the placement of the last customer at stage \( j \) in between customers \( i \) and \( i' \). In this case, customers \( i \) and \( i' \) must have had a precedence relationship at stage \( j - 1 \) which refers to the variable \( o_{j-1,i,i'} \) in the right hand side of the constraint. Note that variables \( o_{j,i,j} \) and \( o_{j,i,i'} \) cannot be 1 at the same time. Thus, the constraint (5.41) along with constraints (5.19)-(5.23) ensure feasible sequences at each stage. In Section 5.4 we present numerical results of the experiments to test the performance of the solution method with this alternative formulation.

5.3.2 Solution to Scenario Subproblems

Since our model has complete recourse, given a first stage solution with a feasible sequence and feasible appointment times and allowances, the optimal second stage solution can be computed directly by calculating the corresponding waiting times, idle time, and overtime variables. That is, given a first stage solution (sequence and appointment times) the scenario subproblem collapses into the same subproblem structure presented in Chapter 3 which can be easily solved recursively. For instance, assuming that a first stage solution to a 3 customer problem is given, and the sequence according to this solution is \( 2 - 3 - 1 \). Table 5.2 includes the sequences and corresponding second stage variables at each of the 3 decision stages according to this first stage solution. In our implementation of the integer L-shaped method, at each iteration, for each scenario, the subproblem solution is obtained as described above and then used to obtain the optimal dual solution directly. This eliminates the need to solve the subproblem LP (e.g. using the simplex method). Thus, much less computational effort is expended in computing the optimal solution to the second stage LP. Note that, in contrast to the dynamic scheduling
model (D-ASP) of Chapter 4, a closed form expression is not available for the dual solution of the recourse problem, which is needed to compute the optimality cuts. However, the optimal basis of the primal problem can be used to efficiently compute the dual solution.

Table 5.2: Second stage variables for a 3 customer problem given a first stage solution for sequencing decisions is known

<table>
<thead>
<tr>
<th>Stage</th>
<th>Sequence</th>
<th>Corresponding Second Stage Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0 − 1 − 2 (0 and 2 dummy)</td>
<td>[ w_{1,0,1} = (Z_0 - x_{1,0,1})^+ ] [ w_{1,1,2} = (w_{1,0,1} + Z_1 - x_{1,1,2})^+ ] [ s_{1,0,1} = (x_{1,0,1} - Z_0)^+ ] [ s_{1,1,2} = (x_{1,1,2} - w_{1,0,1} - Z_1)^+ ] [ \ell_1 = (s_{1,0,1} + s_{1,1,2} + Z_0 + Z_1 - x_{1,0,1} - d)^+ ]</td>
</tr>
<tr>
<td>2</td>
<td>0 − 2 − 1 − 3 (0 and 3 dummy)</td>
<td>[ w_{2,0,2} = (Z_0 - x_{2,0,2})^+ ] [ w_{2,1,3} = (w_{2,0,2} + Z_2 - x_{2,2,1})^+ ] [ w_{2,1,3} = (w_{2,2,1} + Z_1 - x_{2,1,3})^+ ] [ s_{2,0,2} = (x_{2,0,2} - Z_0)^+ ] [ s_{2,1,3} = (-w_{2,2,1} - Z_1 + x_{2,1,3})^+ ] [ \ell_2 = (s_{2,0,2} + s_{2,1,3} + Z_0 + Z_1 + Z_2 - x_{2,0,2} - d)^+ ]</td>
</tr>
<tr>
<td>3</td>
<td>0 − 2 − 3 − 1 − 4 (0 and 4 dummy)</td>
<td>[ w_{3,0,2} = (Z_0 - x_{3,0,2})^+ ] [ w_{3,2,3} = (w_{3,0,2} + Z_2 - x_{3,2,3})^+ ] [ w_{3,1,4} = (w_{3,2,3} + Z_3 - x_{3,1,4})^+ ] [ s_{3,0,2} = (-Z_0 + x_{3,0,2})^+ ] [ s_{3,2,3} = (-w_{3,0,2} - Z_2 + x_{3,2,3})^+ ] [ s_{3,1,4} = (-w_{3,2,3} - Z_3 + x_{3,1,4})^+ ] [ \ell_3 = (s_{3,0,2} + s_{3,2,3} + s_{3,1,4} + Z_0 + Z_1 + Z_2 + Z_3 - x_{3,0,3} - d)^+ ]</td>
</tr>
</tbody>
</table>

5.3.3 Big M Values

Both first stage and second stage problems given in Section 5.2 have big M values in the formulations. These values must be chosen very carefully because having unnecessarily large M values can cause computational disadvantages in solving mixed integer programs (MIPs) since they lead to a weak LP relaxation.

In our formulation, we used big M values for an upper bound on the values of first stage decision variables, allowances, \( x \), and appointment times, \( a \), and second stage decision variables,
waiting time, w and idle time, s. To tighten the formulation we developed upper bounds on the big M values. For the first stage constraints, (5.28) and (5.29), we let

\[ M_1 = \sum_{i=1}^{n} \bar{Z}_i \]  

where \( \bar{Z}_i \) denotes the average service duration for patient \( i \) across all scenarios. This is a valid bound because none of the allowances or the appointment times can take a value bigger than the sum of the durations of all customers in the optimal solution.

We tighten the bounds on \( M_2(\omega) \) and \( M_3(\omega) \) on the second stage variables w and s similarly. Note that \( M_2(\omega) \) and \( M_3(\omega) \) are scenario dependent. Thus, bounds on \( M_2(\omega) \) and \( M_3(\omega) \) can take advantage of the individual scenario, \( \omega \). We use the fact that none of the waiting time variables can take values larger than the sum of the service durations of all customers for each scenario, \( \omega \). This is true since it is not possible for a customer to wait more than the sum of completion times of all customers. This bound can be tightened further by making the bound customer specific. Since each customer \( i \)'s waiting time must be less than or equal to the total service durations of other customers the new bound can be achieved as follows by setting

\[ M_2(\omega) = \sum_{i=1,i \neq i'}^{n} Z_i(\omega). \]  

The allowance for each customer is bounded above by the sum of service durations of all customers because the idle time between two customers, say \( i \) and \( i' \), can never exceed this bound minus the duration of the customer \( i \) (given that \( i \) precedes \( i' \)). Thus \( M_3(\omega) \) can be written as follows:

\[ M_3(\omega) = \sum_{k=1,k \neq i}^{n} Z_k(\omega) - Z_i(\omega). \]  

### 5.3.4 Valid Inequalities

To improve convergence, new cuts are added to the first stage problem. These cuts are constructed in a similar way to the (VI-3) in Section 3. The goal is to provide a tighter bound on theta in the first stage problem using the mean value problem. To construct this mean value based subproblem, the random scenario duration, \( Z_i \), are replaced with their mean value, \( \mu_i \), in a single scenario subproblem. By Jensen’s inequality, the solution to this mean value subproblem provides a lower bound on the value of the recourse problem, thus on the value of \( \theta \). Then new auxiliary variables, \( \bar{w}, \bar{s} \) and \( \bar{l} \), which represent the waiting time, idle time
and overtime variables in this mean value based subproblem. The cuts based on mean value relaxation are as follows:

\[
\theta \geq \sum_{j=1}^{n} \sum_{i=1}^{j} \sum_{i'=1}^{j} \left( c_{i}^{w} \bar{w}_{j,i,i'}(\mu) + c_{i}^{s} \bar{s}_{j,i,i'}(\mu) + c_{i}^{\ell} \bar{\ell}_{j}(\mu) \right) + \sum_{i, i', j} \left( c_{i}^{w} \bar{w}_{j,i,i'}(\mu) \leq M \bar{o}_{j,i,i'}(\mu) \right) \leq 0 \quad \forall i, i', j \tag{5.46}
\]

\[
\bar{w}_{j,i,i'}(\mu) \leq M \bar{o}_{j,i,i'}(\mu) \quad \forall i, i', j \tag{5.47}
\]

\[
\bar{s}_{j,i,i'}(\mu) \leq M \bar{o}_{j,i,i'}(\mu) \quad \forall i, i', j \tag{5.48}
\]

\[
\bar{\ell}_{j}(\mu) \geq \sum_{i=1}^{j} \sum_{i'=1}^{j} \bar{s}_{j,i,i'}(\mu) + \sum_{i=1}^{j} \mu_{i} + \sum_{i'=1}^{j} x_{j,i,i'} - d \quad \forall j \tag{5.49}
\]

These new cuts are added to the first stage problem to provide a better lower bound. Experiments on the performance of this new formulation are presented in Section 5.4.1.

5.3.5 Integer L-Shaped Method

We solved the (D-ASSP) using the integer L-shaped Method. The integer L-shaped method is an iterative method, which proceeds by improving the approximation of the first stage problem (master problem), similar to the L-shaped method described in Chapter 2 [8]. However, instead of solving an LP in the master problem, a MIP solved. After finding an integer feasible solution to the master problem, all subproblems are solved and a new optimality cut is generated from the dual solutions of the subproblems. The optimality cut is added to improve the master problem solution which is subsequently re-solved. This continues until the same stopping criteria have been met.

In our experiments, we test several standard ways to improve the solution performance of the MIP in the first stage problem. For instance, we utilize the “presolve” option of CPLEX 11.0 which eliminates redundant variables and constraints. We also tested the “warm start” option of CPLEX by using the optimal solution of the MIP in the master problem of the previous iteration as a starting solution for the MIP in the current iteration’s master problem.

We experimented with adding many types of MIP cuts. We added generalized upper bound cover cuts and implied bound cuts. A generalized upper bound cover cut is a constraint in which a sum of binary decision variables is less than or equal to 1. Implied bound cuts are
defined as cuts to reflect the bounds of binary variables on continuous variables. Both of these cuts are discussed in [93].

We evaluated the solution performance with several search techniques and variable selection methods offered by CPLEX within the branch and bound. We observed that using different search techniques such as traditional branch and cut or CPLEX’s dynamic search, often provides feasible and optimal solutions more quickly than branch and cut. We also tested different variable selection strategies, such as strong branching, which selects the variables based on the most promising branch to reduce the number of nodes to explore in the solution process.

Our implementation of the integer L-shaped algorithm is summarized in the following pseudocode:

**Integer L-Shaped Algorithm**

1. \( \nu = 1 \) (iterations), \( \omega = 1 \) (scenario), initialize \( M_1 \), \( M_2(\omega) \) and \( M_3(\omega) \), \( \forall \omega \)
2. Initialize L-shape tolerance = 0.01 and tolerance reduction factor , \( \beta \)
3. if option=0
4. Use formulation in (5.19)-(5.31) for the master problem
5. else if option=1
6. Add mean value based cuts to the master problem
7. if option=2
8. Use the alternative formulation with (5.41) for the master problem
9. else if option=3
10. Add mean value based cuts to the master problem with alternative formulation
11. Initialize optimality tolerance for MIP solver
12. While \((\text{L-shape gap} > \text{L-shape tolerance}) \text{ and } (\text{Current time} < \text{Time limit})\) do
13. \( \nu \leftarrow \nu + 1 \)
14. L-shape tolerance = L-shape gap / \( \beta \)
15. Solve master problem to optimality and find an integer feasible solution
16. Solve subproblem for each scenario \( \omega \)
17. Add optimality cut to the master problem
18. Turn CPLEX Presolve option on
19. Turn CPLEX options on to add CPLEX MIP cuts to the master problem
20. Update L-shape gap= 100 (best obj. value - current obj. value)/(best obj. value)
In our implementation, we used a dynamic strategy to update the optimality tolerance, which is referred as L-shape tolerance above, for the MIP solver. We start with a large initial tolerance (1%) and decrease it dynamically at each iteration of the integer L-shaped method. This reduces time spent optimizing the MIPs in the early iterations when the approximation of the recourse function is poor. The L-shape gap is a percentage that is calculated as the ratio of the difference between best objective function value found and current objective function value to the best objective function value found.

5.4 Results

In this section, we first present results illustrating the structure of the optimal sequence and schedule for varying scheduling environments and customer types for different size of problems. Next, we present the results of the experiments to compare the computational performance of the model with and without adding the mean value based cuts. Our initial experiments showed that alternative formulation given in Section 5.3.1 for the first stage problem provides more efficient solution performance than the formulation presented in Section 5.2. Thus, all of the results presented in the next section are based on the experiments with the alternative formulation.

5.4.1 Structure of the Optimal Solution

Experiments to evaluate the structure of the optimal sequence and schedule were conducted considering problem instances with 5 customers. In the first instance, all customers are assumed to have identical cost coefficients for waiting times \((c^w_i = 4, \forall i)\) and appointment times \((c^a_i = 2, \forall i)\), and probability of requesting appointments \((p_i = 0.5, \forall i)\), and service time distributions \((Z_i \sim U(30, 40), \forall i)\). The cost of overtime is set be \(c^\ell = 10\), and cost of provider’s idle time is \(c^s = 5\). The optimal sequence and appointment times for this problem are presented in Table 5.3. The customers are written in bold face in the sequences and dummy customers are written in regular font at the beginning and at the end of the sequences. Results in Table 5.3 indicate that when all of customers are identical in terms of service durations and cost coefficients, FCFS is optimal. Note that this is consistent with Proposition 7 of Section 5.1.

The problem instance presented above has a considerable amount of symmetry due to identical characteristics of customers. Therefore there exist multiple optimal solutions with different customer sequences. From Table 5.3 we see that FCFS is optimal for this problem; however, it is not the only optimal solution. Table 5.4 includes an alternative solution to the same problem.
Table 5.3: Optimal solution of a 5 customer problem instance with identical patient characteristics, $c_i^w = 4, c_i^a = 2, p_i = 0.5, \forall i, c^f = 10$, $c^s = 5, Z_i \sim U(20, 40), \forall i$

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Appointment Times</th>
<th>Allowances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 1: 0 – 1 – 2</td>
<td>$a_1 = 0$</td>
<td>$x_{1-2} = 30.22$</td>
</tr>
<tr>
<td>Stage 2: 0 – 1 – 2 – 3</td>
<td>$a_2 = 30.22$</td>
<td>$x_{2-3} = 33.01$</td>
</tr>
<tr>
<td>Stage 3: 0 – 1 – 2 – 3 – 4</td>
<td>$a_3 = 63.22$</td>
<td>$x_{3-4} = 35.05$</td>
</tr>
<tr>
<td>Stage 4: 0 – 1 – 2 – 3 – 4 – 5</td>
<td>$a_4 = 98.23$</td>
<td>$x_{4-5} = 33.44$</td>
</tr>
<tr>
<td>Stage 5: 0 – 1 – 2 – 3 – 4 – 5 – 6</td>
<td>$a_5 = 131.65$</td>
<td></td>
</tr>
</tbody>
</table>

Note that the values of the appointment times and allowances do not change, however, those values belong to different customers due to having a different optimal sequence.

Table 5.4: Alternative optimal solution of a 5 customer problem instance with identical patient characteristics, $c_i^w = 4, c_i^a = 2, p_i = 0.5, \forall i, c^f = 10$, $c^s = 5, Z_i \sim U(20, 40), \forall i$

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Appointment Times</th>
<th>Allowances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 1: 0 – 1 – 2</td>
<td>$a_5 = 0$</td>
<td>$x_{5-1} = 30.22$</td>
</tr>
<tr>
<td>Stage 2: 0 – 1 – 2 – 3</td>
<td>$a_1 = 30.22$</td>
<td>$x_{1-2} = 33.01$</td>
</tr>
<tr>
<td>Stage 3: 0 – 1 – 2 – 3 – 4</td>
<td>$a_2 = 63.22$</td>
<td>$x_{2-4} = 35.05$</td>
</tr>
<tr>
<td>Stage 4: 0 – 1 – 2 – 3 – 4 – 5</td>
<td>$a_4 = 98.23$</td>
<td>$x_{4-3} = 33.44$</td>
</tr>
<tr>
<td>Stage 5: 0 – 5 – 1 – 2 – 4 – 3 – 6</td>
<td>$a_3 = 131.65$</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.5 presents results of a different 5 customer problem instance which includes two different customer types, 3 routine and 2 add-ons. Routine customers are known to request appointments with certainty. Add-on customers request appointments with probability 0.5. Waiting time cost for routine customers is $c_i^w = 4$, and for add-on customers it is $c_i^w = 8$. Appointment time cost for routine customers is $c_i^a = 2$, for add-on customers is $c_i^a = 6$. This problem instance is motivated by health care environments in which add-on patients have priority over routine patients due to their critical condition. For instance, in surgery scheduling,
urgent add-on patients usually cannot afford to wait, thus, they are scheduled early in the
day. Service distribution, cost of overtime and cost of idle time are the same as the previous
experiments \( (c^\ell = 10, c^s = 5, Z_i \sim U(20, 40), \forall i) \). The optimal sequence, \( o \), appointment
times, \( a \), and allowances, \( x \), are presented in Table 5.6. The results show the optimal sequence
places add-on customers at the beginning of the schedule (if they request appointments) due to
their high cost of appointment times. This indicates that the scheduler should reserve time at
the beginning of the day to accommodate the possibility of urgent add-on customers. Note that
the first routine customer is also scheduled to arrive at time 0 along with the add-on customers
(if they request appointments). Thus, this customer will be served first if the add-on customers
do not request appointments. Also note that for this problem instance, the optimal sequence
is AOFS.

Table 5.5: Optimal solution of a 3 Routine + 2 Add-on customer problem instance, \( c^w_i =
4 \forall i = 1, 2, 3, c^w_i = 8 \forall i = 4, 5, c^a_i = 2 \forall i = 1, 2, 3, c^a_i = 6 \forall i = 4, 5, c^\ell = 10, c^s = 5, p_i = 0.5,\)
\( Z_i \sim U(20, 40), \forall i \)

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Appointment Times</th>
<th>Allowances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 1: 0 – 1 – 2</td>
<td>( a_5 = 0 )</td>
<td>( x_{5-4} = 0 )</td>
</tr>
<tr>
<td>Stage 2: 0 – 1 – 2 – 3</td>
<td>( a_4 = 0 )</td>
<td>( x_{4-1} = 0 )</td>
</tr>
<tr>
<td>Stage 3: 0 – 1 – 3 – 2 – 4</td>
<td>( a_1 = 0 )</td>
<td>( x_{1-3} = 34.43 )</td>
</tr>
<tr>
<td>Stage 4: 0 – 4 – 1 – 3 – 2 – 5</td>
<td>( a_3 = 34.43 )</td>
<td>( x_{3-2} = 33.75 )</td>
</tr>
<tr>
<td>Stage 5: 0 – 5 – 4 – 1 – 3 – 2 – 6</td>
<td>( a_2 = 70.18 )</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.6 provides the optimal schedule of the 3 Routine + 2 Add-on customers problem for
a different choice of cost parameters. In this problem cost coefficients for add-on customers are
changed to \( c^a_i = 4 \) and the probabilities of requesting appointments for the 2 add-on customers
are 0.5 and 0.3 respectively. In the optimal solution, the first add-on customer will be scheduled
to arrive at the beginning of the schedule, and the second add-on customer will be scheduled to
arrive at the end of the daily schedule. This example illustrates the sensitivity of the sequencing
decisions to the indirect waiting costs (\( c^a \)). As the cost of indirect waiting, and the probability
of having customer 5, becomes lower, relative to the example in Table 5.5, we observe a mixed
sequence of routine and add-on customers.

In addition to the above experiments we experimented with cases in which customers have
Table 5.6: Optimal solution of a 3 Routine + 2 Add-on customer problem instance, $c_i^w = 4 \forall i = 1, 2, 3$, $c_i^w = 8 \forall i = 4, 5$, $c_i^a = 2 \forall i = 1, 2, 3$, $c_i^a = 4 \forall i = 4, 5$, $c^f = 10$, $c^s = 5$, $p_i = 0.5 \forall i = 1, 2, 3, 4$, $p_5 = 0.3$, $Z_i \sim U(20, 40), \forall i$

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Appointment Times</th>
<th>Allowances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 1: 0 – 1 – 2</td>
<td>$a_4 = 0$</td>
<td>$x_{4-1} = 0$</td>
</tr>
<tr>
<td>Stage 2: 0 – 1 – 2 – 3</td>
<td>$a_1 = 0$</td>
<td>$x_{1-3} = 34.26$</td>
</tr>
<tr>
<td>Stage 3: 0 – 1 – 3 – 2 – 4</td>
<td>$a_3 = 34.26$</td>
<td>$x_{3-2} = 35.47$</td>
</tr>
<tr>
<td>Stage 4: 0 – 4 – 1 – 3 – 2 – 5</td>
<td>$a_2 = 69.73$</td>
<td>$x_{2-5} = 64.77$</td>
</tr>
<tr>
<td>Stage 5: 0 – 4 – 1 – 3 – 2 – 5 – 6</td>
<td>$a_5 = 134.51$</td>
<td></td>
</tr>
</tbody>
</table>

different variances for their service durations. Previous research indicates that scheduling customers with higher variances later in the schedule minimizes the potential impact of waiting time for the later customers in the schedule [88, 23]. This limits the amount of disruption in the schedule these customers can cause. For this experiment, 5 customers having the same mean duration but different variances are considered. Cost of appointment time, for each customer is fixed to $c_i^a = 0$ in order to prevent its effect on the sequencing decisions. Service durations are as follows: $Z_1 \sim U(25, 35)$, $Z_2 \sim U(15, 45)$, $Z_3 \sim U(20, 40)$, $Z_4 \sim U(23, 37)$, $Z_5 \sim U(10, 50)$. Thus service durations have a mean of 30 for each customer, however variances differ. Table 5.7 includes the results which indicate that for the problems with dynamic arrivals of customers with probability $p_i = 0.5, \forall i$, the customers are always sequenced in FCFS order regardless of the changes in the cost coefficients. This is due to the fact that the probability of having additional customers is low when $p_i = 0.5, \forall i$, thus scheduling a customer which has a very low probability to request an appointment before a higher probability customer is not beneficial. As the conditional probability gets higher ($p_i = 0.9$, and $p_i = 1, \forall i$) in the last two rows of Table 5.7), the uncertainty in appointment requests is reduced and the effect of having variances on the sequence becomes more prominent. Note that $p_i = 1, \forall i$ case refers to the static scheduling problem. In these cases, the optimal sequence indicates scheduling customers with lower variances first. This sequencing rule was defined as “increasing variance rule” (VAR) in Chapter 2. Scheduling lower variance customers first prevents accumulation of waiting later in the schedule which results in lower total cost. The increasing variance rule is proven to be optimal [88] for sequencing 2 customers and presumed to be optimal for 3 customers with uncertain service durations. However, no proof is provided for the problem with more than 3 patients. Furthermore, the results of a simulation study in [88] indicates a sequence starting with customers with
lower variances and then followed by a random order of decreasing variance. Our results are also generally consistent with these findings. However, in the last two examples of Table 5.7 we note some minor variations of the optimal sequence from the VAR ordering.

Table 5.7: Optimal sequencing rules with respect to different cost parameters and conditional probabilities \( p \) in the presence of customers with different variances

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Optimal Sequence</th>
<th>Sequencing Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>( c^w = c^s = c^\ell = 1 ) ( p_i = 0.5 \ \forall \ i = 1, \ldots, 5 )</td>
<td>1 – 2 – 3 – 4 – 5</td>
<td>FCFS</td>
</tr>
<tr>
<td>( c^w = 10, \ c^s = c^\ell = 1 ) ( p_i = 0.5 \ \forall \ i = 1, \ldots, 5 )</td>
<td>1 – 2 – 3 – 4 – 5</td>
<td>FCFS</td>
</tr>
<tr>
<td>( c^w = c^s = 1, \ c^\ell = 10 ) ( p_i = 0.5 \ \forall \ i = 1, \ldots, 5 )</td>
<td>1 – 2 – 3 – 4 – 5</td>
<td>FCFS</td>
</tr>
<tr>
<td>( c^w = 10, \ c^s = c^\ell = 1 ) ( p_i = 0.9 \ \forall \ i = 1, \ldots, 5 )</td>
<td>1 – 3 – 4 – 2 – 5</td>
<td>VAR (with one exception)</td>
</tr>
<tr>
<td>( c^w = 10, \ c^s = c^\ell = 1 ) ( p_i = 1 \ \forall \ i = 1, \ldots, 5 )</td>
<td>1 – 3 – 4 – 2 – 5</td>
<td>VAR (with one exception)</td>
</tr>
</tbody>
</table>

5.4.2 Computational Performance of Proposed Methods

We tested several implementations of the integer L-shaped method to solve (D-ASSP) we utilized several ways to improve the solution performance. Our experiments show that a number of the options we tested do not bring any computational efficiency in solving master problems with large number of customers. For instance, we utilize the presolve option in CPLEX which eliminates redundant variables and constraints. We also tested the warm start option in CPLEX by using the optimal solution of the MIP in the master problem of the previous iteration as a starting solution for the MIP in the current iteration’s master problem. We generally observed that warm start option made little effect when presolve is utilized, and presolve is generally more efficient in solving MIPs.

We also tested the effects of adding many types of MIP cuts. Further, from our initial experiments we observed that adding generalized upper bound cover cuts and implied bound cuts improved solution performance. Aggressive addition of other cuts such as mixed integer rounding cuts, cliques cuts, fractional cuts, flow cover cuts had little or no effect on the solution time.
We observed that using different search techniques such as traditional branch-and-cut or CPLEX’s dynamic search made no significant difference on the solution time. Also, using different variable selection strategies such as strong branching also did not have significant effect on the solution time.

We compare the computational performance of the methods in terms of the number of integer L-shaped method iterations, and the total CPU time to complete those iterations. The first set of experiments compares the performance of the solution methods discussed in Section 5.3.5. Two alternative implementations of the integer L-shaped method are compared. In the first implementation, CPLEX MIP cuts (chosen by CPLEX 11.0’s automated capability) are added to first stage problem. The choice of the cuts added by CPLEX varies with problem instance and from one iteration to another within the solution process of the same instance. The cuts added by CPLEX include flow-cover cuts, zero-half cuts and Gomory cuts. Our preliminary experiments showed that adding GUB cover cuts and implied bound cuts along with the other MIP cuts chosen by CPLEX 11.0 MIP optimizer results in reduced computation time for all model instances. In the second implementation, we included the mean value based cuts presented in Section 5.3 to the first stage problem along with the cuts added by CPLEX 11.0’s MIP optimizer.

All of the model instances used in the experiments of this section were created by sampling using 10 different random seeds, i.e. an instance is replicated 10 times with different random number seeds to generate service duration scenarios. The results are presented in terms of the average and maximum CPU time and number of iterations of the 10 replications for each test instance.

Two different service time distributions are considered for the experiments: uniform and lognormal distribution. As explained in Chapter 4, uniform distribution is considered to represent the service durations in the primary care clinics, and lognormal distribution is considered to represent the service durations in specialty clinics (e.g. endoscopy clinics). The results of the experiments with uniformly distributed service durations are presented in Table 5.8. Instances 2.1, 2.3., and, 2.5 are dynamic scheduling problems including a single customer type with \( Z_i \sim U(30,40) \), \( p_i = 0.5 \), \( c_i^p = 2 \), \( c_i^w = 4 \), \( \forall i \). Instances 2.2, 2.4, and, 2.6 include two customer types, routine and urgent. Routine customers are scheduled with certainty \( (p_i = 1) \) and urgent customers request appointments dynamically with \( p_i = 0.5 \). The cost coefficients for urgent customers are \( c_i^w = 8 \), and \( c_i^a = 6 \). Table 5.9 presents the results of the same experiments with lognormal service durations \( (Z_i \sim Lognormal(3.2,0.5)) \).

The experiments with 5 patients (instances 2.1, 2.2, 3.1 and 3.2) are solved within the predetermined 1% optimality gap. In all of these instances with one exception (instance set 3.2 with lognormal distribution on Table 5.9), adding mean value based cuts resulted in significant computational advantage both in number of iterations and computation time. Only
one set of instances (instance set 2.3) for bigger size problems was solved to optimality within a predetermined time limit of 15000 seconds. For these instances, the results at the time of termination are presented in the tables. The results indicate that as the number of patients gets larger, the problems get really hard to solve. When the number of iterations at the time of termination is considered, we can conclude that the master problems with the mean value cuts take significantly longer time to solve since much lower number of iterations were performed within the same amount time.

In order to make a coherent comparison of the performance, the quality of the solution at the time of termination is also considered. Table 5.10 includes the optimality gap at the time of termination for the instances that could not be solved to optimality. The results presented in the table are the worst, best and average gaps found within 10 replication of the instances with different random seeds. The percentage gap is calculated as the ratio of the difference between best objective function value and current objective function value to the best objective function value found. The results indicate that even though adding mean value cuts complicates the master problem structure, much better solutions are obtained in fewer iterations.
Table 5.8: Computational performances of solution methods with uniformly distributed service times ($Z_i \sim U(30, 40)$). Problems were solved to a tolerance of 1% with a maximum time of 15000 seconds. An asterisk (*) indicates cases in which no model instance were solved to the specific tolerance. Times are reported in seconds.

<table>
<thead>
<tr>
<th>Problem Size (Patients)</th>
<th>Instance No</th>
<th>Patient Class</th>
<th>L-Shaped Method (with CPLEX cuts)</th>
<th>L-Shaped Method (with CPLEX + mean value based cuts)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>CPU Time # of Iterations</td>
<td>CPU Time # of Iterations</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Average Max Average Max</td>
<td>Average Max Average Max</td>
</tr>
<tr>
<td>5</td>
<td>2.1</td>
<td>5 Identical Patients</td>
<td>449 484 192.9 202</td>
<td>161 183 41.7 48</td>
</tr>
<tr>
<td></td>
<td>2.2</td>
<td>3 Routine + 2 Urgent Patients</td>
<td>2247.71 2546 608.7 660</td>
<td>705.5 956 198 261</td>
</tr>
<tr>
<td>7</td>
<td>2.3</td>
<td>7 Identical Patients</td>
<td>15021* 15041* 283 290</td>
<td>4011 4522 53 59</td>
</tr>
<tr>
<td></td>
<td>2.4</td>
<td>4 Routine + 3 Urgent Patients</td>
<td>15033* 15070* 241 247</td>
<td>15132* 15404* 43 45</td>
</tr>
<tr>
<td>10</td>
<td>2.5</td>
<td>10 Identical Patients</td>
<td>15200* 15407* 92 97</td>
<td>16456* 16462* 9 9</td>
</tr>
<tr>
<td></td>
<td>2.6</td>
<td>7 Routine + 3 Urgent Patients</td>
<td>15208* 15535* 93 102</td>
<td>16418* 16426* 9 9</td>
</tr>
</tbody>
</table>
Table 5.9: Computational performances of solution methods with Lognormally distributed service times ($Z_i \sim \text{Lognormal}(3.2, 0.5)$). Problems were solved to a tolerance of 1% with a maximum time of 15000 seconds. An asterisk (*) indicates cases in which no model instance were solved to the specific tolerance. Times are reported in seconds.

<table>
<thead>
<tr>
<th>Problem Size (Patients)</th>
<th>Instance No</th>
<th>Patient Class</th>
<th>L-Shaped Method (with CPLEX cuts)</th>
<th>L-Shaped Method (with CPLEX + mean value based cuts)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>CPU Time</td>
<td># of Iterations</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Average</td>
<td>Max</td>
</tr>
<tr>
<td>5</td>
<td>3.1</td>
<td>5 Identical Patients</td>
<td>1413.38</td>
<td>1773</td>
</tr>
<tr>
<td></td>
<td>3.2</td>
<td>3 Routine + 2 Urgent</td>
<td>1500.11</td>
<td>4161</td>
</tr>
<tr>
<td>7</td>
<td>3.3</td>
<td>7 Identical Patients</td>
<td>15039*</td>
<td>15091*</td>
</tr>
<tr>
<td></td>
<td>3.4</td>
<td>4 Routine 3 Urgent</td>
<td>15043*</td>
<td>15080*</td>
</tr>
<tr>
<td>10</td>
<td>3.5</td>
<td>10 Identical Patients</td>
<td>15185*</td>
<td>15328*</td>
</tr>
<tr>
<td></td>
<td>3.6</td>
<td>7 Routine 3 Urgent</td>
<td>15093*</td>
<td>15188*</td>
</tr>
</tbody>
</table>
Table 5.10: L-shaped gap at the time of termination for the instances that are not solved to optimality

<table>
<thead>
<tr>
<th>Problem Size</th>
<th>Instance No</th>
<th>Patient Type</th>
<th>L-Shaped Method (with CPLEX cuts)</th>
<th>L-Shaped Method (with CPLEX cuts + mean value based cuts)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 (uniform)</td>
<td>2.3</td>
<td>7 Identical Patients</td>
<td>Worst: 116.46% Best: 107.12% Average: 111.95%</td>
<td>optimal</td>
</tr>
<tr>
<td></td>
<td>2.4</td>
<td>4 Routine 3 Urgent</td>
<td>Worst: 179.60% Best: 174.62% Average: 177.8%</td>
<td>Worst: 2.03% Best: 1.95% Average: 2.01%</td>
</tr>
<tr>
<td>10 (uniform)</td>
<td>2.5</td>
<td>10 Identical Patients</td>
<td>Worst: 269.08% Best: 240.11% Average: 253.07%</td>
<td>Worst: 8.51% Best: 7.26% Average: 7.41%</td>
</tr>
<tr>
<td></td>
<td>2.6</td>
<td>7 Routine 3 Urgent</td>
<td>Worst: 400.49% Best: 375.32% Average: 386.86%</td>
<td>Worst: 2.06% Best: 1.99% Average: 2.02%</td>
</tr>
<tr>
<td>7 (lognormal)</td>
<td>3.3</td>
<td>7 Identical Patients</td>
<td>Worst: 344.27% Best: 223.32% Average: 252.53%</td>
<td>Worst: 26.07% Best: 21.99% Average: 24.60%</td>
</tr>
<tr>
<td></td>
<td>3.4</td>
<td>4 Routine 3 Urgent</td>
<td>Worst: 467.83% Best: 335.02% Average: 371.82%</td>
<td>Worst: 16.57% Best: 15.71% Average: 15.99%</td>
</tr>
<tr>
<td>10 (lognormal)</td>
<td>3.5</td>
<td>10 Identical Patients</td>
<td>Worst: 376.86% Best: 338.37% Average: 349.87%</td>
<td>Worst: 33.69% Best: 31.53% Average: 33.02%</td>
</tr>
<tr>
<td></td>
<td>3.6</td>
<td>7 Routine 3 Urgent</td>
<td>Worst: 561.59% Best: 517.30% Average: 532.67%</td>
<td>Worst: 13.53% Best: 13.07% Average: 13.33%</td>
</tr>
</tbody>
</table>

The results that we present in this section are for the problem instances that are particularly difficult to solve. For instance, having the same the cost parameters for the customers within the same customer class produces a significant amount of symmetry. To show this we tested new instances in which first stage costs, $c_d^i$, and second stage waiting time cost, $c_w^i$, are different for each customer. Table 5.11 includes results for these experiments for the same size problems presented in Tables 5.8 and 5.9. All of the instances are solved to optimality within the time limit.

### 5.5 Conclusions

In this chapter, we fist provided motivation for the scheduling problems for which FCFS discipline may not be optimal. We relaxed the FCFS assumption and proposed a general dynamic (on-line) appointment scheduling model which also includes sequencing of customers. We formulated the dynamic appointment sequencing and scheduling problem as a two-stage stochastic integer program. We provided two alternative formulations for the fist stage problem which produces a feasible sequence of customers. We used two alternative implementations of the
integer L-shaped method to solve this problem, including one that uses cuts based on the mean value problem. We presented the results of numerical experiments using these two implementations to compare computational performance. We also provide results for a variety of model instances that give general insights into the structure of optimal decisions for on-line appointment scheduling.

Our results indicate that adding mean value based cuts to the master problems produced computational efficiency in most of the instances (instance number 2.1, 2.2, and 3.1) that were solved within the time limit. We observed that as the problem size becomes larger (7 and 10 patient model instances), none of the instances were solved to the optimality tolerance of 1% within the 15000 seconds time limit, with or without the mean value based cuts. However, we observed that adding mean value based cuts to the master problems produced significant improvements in the optimality gap. For instance, the smallest gap at the time of termination for the 7 customer model instances (instances 2.3, 2.4, 3.3, and 3.4) without mean value cuts was 107.2%, whereas the same set of instances were solved to the optimality tolerance when the mean value cuts are added. For the model instances with 10 customers (instances 2.5, 2.6, 3.5, and 3.6), the best gap found without mean value cuts was 240.11%, whereas with mean value cuts, some instances terminated with optimality gaps as small as 2%. Thus, we conclude that greater confidence can be achieved when the mean value based cuts are included in the master problem.

Our results show that, having different customer types effects the sequencing decisions in dynamic appointment scheduling. In the presence of different patient types, different cost parameters, $c^u$, $c^a$, $c^l$ and different appointment request probabilities play an important role in sequencing and scheduling decisions. We found that, when all customers are identical, sequencing is effected by the appointment request probability, $p_i$. For instance, we found that for a dynamic scheduling problem, in which each customer appointment with $p_i = 0.5$, the

Table 5.11: Results for the problem instances that are solved to optimality. ($c^l = 10, c^a = 5, c^u_i = i, c^l_i = i^2$. $Z_i \sim U(30, 40)$ for uniformly distributed service durations, $Z_i \sim LogN(3.2, 0.5)$ for lognormally distributed service durations.)

<table>
<thead>
<tr>
<th>Problem Size</th>
<th>Distribution Type</th>
<th>L-Shaped Method (with CPLEX + mean value cuts)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>CPU Time</td>
</tr>
<tr>
<td>7 Customers</td>
<td>Uniform</td>
<td>41.17</td>
</tr>
<tr>
<td></td>
<td>Lognormal</td>
<td>436.65</td>
</tr>
<tr>
<td>10 Customers</td>
<td>Uniform</td>
<td>215.57</td>
</tr>
<tr>
<td></td>
<td>Lognormal</td>
<td>322.16</td>
</tr>
</tbody>
</table>

87
sequence always indicates FCFS regardless of the cost coefficients. From these results, we can conclude that as the probability of having more customers in the schedule decreases, scheduling those additional customers at the beginning of the day, or, in other words, reserving time for those additional customers, is not optimal. Doing otherwise may produce idle time since there is a high probability that these customers will not request appointments. We observed that as $p_i$ increases, i.e. the uncertainty level in customer requests decreases, the customers with lower variances are scheduled early in the schedule. Scheduling customers in increasing variance order is found to be optimal for the model instances we consider when the problem becomes a static scheduling problem ($p_i = 1$).
Chapter 6

Conclusions

Appointment scheduling systems provide the interface between customer demand and service provider availability; thus, they should balance the needs of both stakeholders. In this thesis we study appointment scheduling problems that involve finding optimal or near optimal appointment times for customer arrivals to a single stochastic server given sources of uncertainty in both service time and customer demand. The total cost of scheduling decisions is often measured by the cost of expected customer waiting time, server idle time, and overtime. The overarching goal of this research was to provide new stochastic optimization models and solution approaches for appointment scheduling problems under service time duration and demand uncertainty. The problems considered were motivated by problems commonly faced in healthcare systems; however, the models and methods can be applied to many types of appointment-based service systems.

In this thesis, we have proposed novel stochastic programming models for several variants of appointment scheduling problems. We first proposed a 2-SLP model for a static appointment scheduling problem in the presence of customer no-shows. Then, we proposed a novel M-SLP model for dynamically scheduling appointments under the assumption of FCFS sequence of customer appointment requests. Finally, we relaxed the FCFS assumption and proposed a new dynamic (on-line) appointment scheduling model which finds optimal schedule as well as the sequencing of customers. In each case, we have proposed several methods that take advantage of the underlying model structures and build on the existing stochastic programming literature.

In the remainder of this chapter, we summarize the most significant conclusions that can be drawn from Chapters 3, 4, 5 and we summarize opportunities for future research.
6.1 Thesis Contributions

The models and solution methods presented in this thesis are novel approaches to extensions of well known problems which have received attention over the past 30 years. Previous research focused on using queuing methods, considering exponential service times and steady state behavior, which often do not reflect health care environments. Heuristics and simulation methods have also been studied to find better scheduling rules for the appointment systems. Our models contribute the literature by providing optimal or near optimal solution to these complicated appointment scheduling problems. In the case of dynamic appointment scheduling, to our knowledge, we are the first to present optimal on-line appointment schedules.

The 2-SLP model presented in Chapter 3, (NS-ASP), aims to solve the appointment scheduling problem in the presence of service duration uncertainty and customer no-shows. Previous research in the medical literature has focused on identifying the root causes of patient no-shows, and proposing interventions to decrease no-show rates (i.e. phone calls, reminder cards). Research on operations research/management science literature, on the other hand has mostly focused on comparing different scheduling policies (i.e. overbooking to compensate no-shows) on the system’s overall performance with using queuing theory or simulation methods. Our work focuses on finding the optimal schedules for different scheduling environments with varying cost coefficients and no-show rates. Our findings show that the optimal schedule is quite sensitive to no-show rates as well as the relative importance of customer waiting time as opposed to service provider’s overtime. The results of our experiments indicate that for environments with high no-show rates, one or more customers should be scheduled to arrive at the same time (double-booking). Our findings indicate that double-booking occurs especially when provider’s overtime cost is significantly higher than customer waiting time cost (a factor of 10 or more). We have also provided insights on the effects of no-shows on the financial performance of the scheduling systems by comparing the total cost of the schedule considering the expected number of customers in the system. For instance, in the example we consider, scheduling 10 customers in the presence of no-shows with probability $p = 0.2$ indicates an optimal schedule which is 38% costlier than the schedule obtained when no-shows do not exist. This percentage increases to 126% as the no-show rate increases from 0.2 to 0.3. We concluded that under the optimal schedules, in the presence of no-shows, total cost is very sensitive to customer probabilities of no-show.

The M-SLP model we presented in Chapter 4, (D-ASP), considers the dynamic nature of the appointment systems. In reality, appointments are often assigned dynamically, one at a time, as the customers request appointments. We introduced test instances that were motivated by scheduling environments encountered in health care systems (e.g. endoscopy suite). We considered two different type of patients; routine patients, who are known to be scheduled,
and add-on patients, who request appointments probabilistically. We found that the structure of the optimal allowances follow a dome-shape, which has been observed to be optimal for static scheduling problems, for the routine patients. However, this pattern is broken for add-on patients. We found that when the overtime is costlier than waiting time, as the uncertainty increases in customer demand, allowances become closer to each other for routine patients, but farther from each other for add-on patients.

We studied the structure of the multi-stage stochastic linear program for (D-ASP) and implemented many ideas to improve convergence. We discovered that the master problems can be solved without the simplex algorithm due to its 2-variable structure. We also eliminated the need for solving a linear program for the subproblems and calculated the dual solutions recursively. We tested a special multi-cut version of nested decomposition which produced a considerable amount of efficiency both in number of iterations and CPU time. Our results indicate that the multi-cut implementation can decrease the number of iterations and total CPU time as much as 50%. The valid inequalities that we developed did not bring significant reduction in number of iterations or CPU time, however, they substantially improved the bound on the optimal solution at early iterations. This latter fact was evidence that such valid inequalities would be effective extensions to discrete optimization problems.

In Chapter 5, we presented (D-ASSP) which relaxes the FCFS assumption in our previous scheduling models. We motivated (D-ASSP) by showing that, for some scheduling environments, especially in health care systems with urgent appointment requests for add-on patients, FCFS is not necessarily the optimal solution. The results of our experiments indicate that, optimal sequencing and the scheduling decisions are sensitive to both the cost parameters and the probability of appointment requests. The structure of the optimal time allowances have similar characteristics found in the previous model (D-ASP); however, the customer sequence varies depending on the relative importance of customers and the variance of their service durations. For instance, our results indicate that, customers with higher appointment time cost, i.e. higher importance, are sequenced at the beginning of the schedule. This pattern breaks down as the probability of appointment requests of these higher priority customers decreases. We also draw insights from the optimal sequencing decisions when customers have different service duration variances. According to our results, as the appointment request probability increases, for example, from $p_i = 0.5$ to $p_i = 0.9$, or to $p_i = 1$, customers with lower variances are often scheduled at the beginning of the schedule. Thus, we conclude that there is an important trade off between the customer service time distribution and their probability of requesting appointments.

Our numerical experiments with (D-ASSP) showed that this problem is computationally challenging due to difficulty in solving the master problem. It appears to have a poor LP relaxation leading to difficulty in solving the MIP via branch-and-bound. In addition, the model
instances of (D-ASSP) that were studied in Chapter 5 are especially hard due to the presence of symmetry between the customers having the same cost coefficients and appointment request probabilities. For example, cases in which customers in the same patient class, i.e. routine and add-ons are indistinguishable. We tested several approaches to solve (D-ASSP) and ultimately developed two alternative implementation of the L-shaped method for stochastic MIPs. We found that adding mean value based cuts produces significantly better bounds on the optimal solution.

6.2 Future Directions

We study scheduling of appointment based service systems in the presence of two complicating factors: customer no-shows and dynamic appointment requests. We evaluated the effects of both of these factors independently on the optimal scheduling decisions. However, in most scheduling environments both of these factors exist simultaneously. For instance, in clinical environments, especially with open-access scheduling, appointments are assigned dynamically and no-shows occur with a significant rate. In surgery scheduling, on the other hand, no-shows often occur in the form of cancelations, and schedules must accommodate urgent add-on surgeries. A more realistic model which incorporates both no-shows and the dynamic nature of the appointment scheduling problem can be developed based on our current models. Such models could include multiple stochastic servers (e.g. ORs), and perhaps multiple stages of service (e.g. surgery followed by recovery). With more general models, the simultaneous effects of demand uncertainty from no-shows and add-on cases could be better estimated.

Our models NS-ASP and D-ASP provides optimal solutions to daily appointment schedules. Our findings from these models can be applied in a multi-day scheduling environment via simulation. For instance the optimal scheduling policies we found under different demand uncertainty can be tested via a realistic simulation model of an outpatient clinic and the effects on the system performance in a multi-period environment can be observed.

Our experiments with solving the dynamic sequencing and scheduling problem (D-ASSP) showed us that these problems are very hard to solve to optimality. Solving the master problem becomes harder especially in the presence of symmetry between customers having same cost coefficients and appointment request probabilities. One way to improve the solution performance may be to introduce additional cuts to eliminate this symmetry. These additional cuts can be used to enforce a fixed sequence within these identical type of customers. Alternative formulations which provide better LP relaxation may also bring computational efficiency in solving the sequencing problem in the first stage. As an alternative to optimizing the first stage problem, heuristic approaches which guarantee a good approximation to the optimal solutions could be explored.
Another future direction to achieve more realistic representation of the scheduling systems is to consider multi-server environments. Most service systems operate with multiple servers but scheduling is done in a centralized way by a single scheduler. Based on our experiments with optimizing a single server system, we can presume that optimization of multi-server appointment systems will be a very challenging problem. Thus, more work is needed to draw theoretical insights to find ways to improve solution performance. The results of Chapters 4 and 5 provide a foundation for taking advantage of the problem structure for multi-server settings.
REFERENCES


Appendix A: Efficient Algorithm to solve 2-Variable Linear Programs

Following is pseudo code for the adaptation of the algorithm proposed by Dyer [29] to solve 2-variable linear programs.

$k$: cut index  
$a_k$: slope of cut $k$  
$b_k$: rhs of cut $k$  
$n$: number of undominated cuts  
$intersect_{i,j}$: x coordinate of intersection of cuts $i$ and $j$

Initialize: Set $x = 0, \theta = 0$  
Sort the lines in increasing slope ($a_k$)  
Check if lines are parallel and delete dominated lines  
Find the solution $x$ and $\theta$:

if $(n = 1)$

if ($a_0 = 0$)  
  \{ $x = 0$, $\theta = \max\{0, b_0\}$ \}
else if ($a_0 \leq 0$)  
  \{ $x = \max\{0, \frac{b_0}{a_0}\}$, $\theta = 0$ \}
else  
  \{ $x = \max\{0, \frac{b_0}{a_0}\}$, $\theta = \max\{0, b_0\}$ \}

if $(n = 2)$

if ($a_0 \leq 0$ & $a_1 \leq 0$)  

if ($a_1 \neq 0$)  
  \{ $x = \max\{0, \frac{b_0}{a_0}, \frac{b_1}{a_1}\}$, $\theta = 0$ \}
else  
  \{ if ($b_1 > b_0$)  
    \{ $x = 0$, $\theta = b_1$ \}
  else  
    \{ if ($b_1 \leq 0$)  
      \{ $x = \max\{0, \frac{b_0}{a_0}\}$, $\theta = 0$ \}
    else  
      \{ $x = \max\{0, \frac{b_0}{a_0}\}$, $\theta = b_1$ \} \}
  \}
else  
  \{ if ($a_0 > 0$ & $a_1 > 0$)  
    \{ $x = 0$, $\theta = \max\{0, b_0, b_1\}$ \}
  else  
    \{ $x = \max\{0, intersect_{0,1}\}$, $\theta = \max\{0, b_0, b_1\}$ \}
  \}

}
if \((n > 2)\) {
    \(\text{solution} = 0\)
    While (solution = 0) {
        if \((a_{n-1} \leq 0)\) { \(\theta = 0, x = \max \{\frac{b_0}{a_0}, \frac{b_1}{a_1}, \ldots, \frac{b_{n-1}}{a_{n-1}}\}\) }
        if \((a_0 \geq 0)\) { \(x = 0, \theta = \max \{b_0, b_1, \ldots, b_{k-1}\}\) }
        else {
            \(\text{Divide } n \text{ lines into } \left\lfloor \frac{n}{2} \right\rfloor \text{ pair of lines}\)
            \(\text{Find pairwise intersections } intersect_{i,j}\)
            \(\text{Find } x_{\text{median}} \text{ of pairwise intersection points}\)
            \(\theta_{\text{max}} = \max_{0 \leq i \leq (k-1)} \{a_i x_{\text{median}} + b[i]\}\)
            \(a_{\text{min}} = \min_{0 \leq i \leq (k-1)} \{a_i \mid a_i x_{\text{median}} + b_i = \theta_{\text{max}}\}\)
            \(a_{\text{max}} = \max_{0 \leq i \leq (k-1)} \{a_i \mid a_i x_{\text{median}} + b[i] = \theta_{\text{max}}\}\)
            if \((a_{\text{min}} \leq 0 \& a_{\text{max}} \geq 0)\) { \(x = x_{\text{median}}, \theta = \theta_{\text{max}}\) }
            else if \((a_{\text{min}} > 0)\) {
                \(x \leq x_{\text{median}}\)
                \(\text{Find intersection points such that } intersect_{i,j} \geq x_{\text{median}}\)
                \(\text{Eliminate line } j \text{ (bigger slope line) from the problem}\) }
            else if \((a_{\text{max}} < 0)\) {
                \(x \geq x_{\text{median}}\)
                \(\text{Find intersection points such that } intersect_{i,j} \leq x_{\text{median}}\)
                \(\text{Eliminate line } i \text{ (smaller slope line) from the problem}\) }
        }
    }
}