8-2014

Large-Scale Solution Approaches for Healthcare and Supply Chain Scheduling

Ridvan Gedik
University of Arkansas, Fayetteville

Large-Scale Solution Approaches for Healthcare and Supply Chain Scheduling

Large-Scale Solution Approaches for Healthcare and Supply Chain Scheduling

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in Industrial Engineering

by

Ridvan Gedik
Middle East Technical University
Bachelor of Science in Industrial Engineering, 2009
University of Arkansas
Master of Science in Industrial Engineering, 2011

August 2014
University of Arkansas

This dissertation is approved for recommendation to the Graduate Council.

Dr. Chase Rainwater Dissertation Director	Dr. Edwin Romeijn Committee Member
Dr. Ronald Rardin Committee Member	Dr. Shengfan Zhang Committee Member

Abstract

This research proposes novel solution techniques for two real world problems. We first consider a patient scheduling problem in a proton therapy facility with deterministic patient arrivals. In order to assess the impacts of several operational constraints, we propose single and multi-criteria linear programming models. In addition, we ensure that the strategic patient mix restrictions predetermined by the decision makers are also enforced within the planning horizon. We study the mathematical structures of the single criteria model with strict patient mix restrictions and derive analytical equations for the optimal solutions under several operational restrictions. These efforts lead to a set of rule of thumbs that can be utilized to assess the impacts of several input parameters and patient mix levels on the capacity utilization without solving optimization problems. The necessary and sufficient conditions to analytically generate exact efficient frontiers of the bicriteria problem without any additional side constraint are also explored. In a follow up study, we investigate the solution techniques for the same patient scheduling problem with stochastic patient arrivals. We propose two Markov Decision Process (MDP) models that are capable of tackling the stochasticity.

The second problem of interest is a variant of the parallel machine scheduling problem. We propose constraint programming (CP) and logic-based Benders decomposition algorithms in order to make the best decisions for scheduling nonidentical jobs with time windows and sequence dependent setup times on dissimilar parallel machines in a fixed planning horizon. This problem is formulated with (i) maximizing total profit and (ii) minimizing makespan objectives. We conduct several sensitivity analysis to test the quality and robustness of the solutions on a real life case study.

Acknowledgments

I owe my gratitude to a great many people who have made the production of this dissertation possible and because of whom I will cherish my graduate school memories forever. First and foremost, I cannot thank my advisor Dr. Chase Rainwater enough for believing in me. The chance to work and study with him is one of the greatest privileges of my life. I have been incredibly fortunate to have him as an advisor who challenged me, kept me enthusiastic and gave me just the right level of freedom and guidance I needed to explore on my own. I am forever grateful for his great mentorship, endless personal and professional support.

I would like to thank my committee members, Drs. Edwin Romeijn, Ronald Rardin and Shengfan Zhang for their guidance, encouragements and invaluable advices throughout my doctoral studies. Their feedbacks have taught me great lessons and made this dissertation more solid. I am also grateful to all my professors for consistently sharing their knowledge without any hesitation to enable me succeed in my studies. I specifically thank Drs. Edward Pohl and Heather Nachtmann for their support and collaboration.

I am thankful to all my friends who made Fayetteville a second home for the past five years. Specifically, I express my sincere gratitudes to Ahmet Aktürk, Emre Kıraç, Furkan Öztanrıseven, Steve Sharp, Serdar Kılınç, Kaan Okyay, Ben Riley, Ryan Black, Payam Parsa, Dia St. John and Jingjing Tong for the great memories. I gladly express my sincere gratitude to the wonderful Industrial Engineering staff, Karen Standley, Tamara Ellenbecker, Carrie Pennington and Sandy Sehon for their tremendous help.

Finally, none of my success would have been possible without the love of my family. I would like to thank my beloved parents Şerife & Dursun Gedik for their continuous prayers and unconditional support during all phases of my life. My sincere gratitudes go to my sister Kübra, my brothers & their wives Numan, Nuh, Fatma and Reyhan for their generous support. I thank Mehmet Oğuz, Bahadır and Zeynep, the youngest members of our family, for sharing their joy with their uncle.

Dedication

To my parents, sister and brothers

Contents

List of Figures

List of Tables

1. INTRODUCTION

This dissertation considers class of problems that seek best scheduling decisions for various considerations and applications with and without uncertainty. In general, scheduling problems have been widely studied over recent decades. In this research, we contribute to this field by considering newly emerged problems in different environments with several variants. We first study patient scheduling problem in a proton therapy facility with deterministic demand. We propose a single and multi-criteria linear programming models in order to capture the impacts of several operational restrictions and then study the problem structures to analytically derive optimal solutions in the case of certain input parameters. Secondly, we relax the deterministic patient arrival assumption in proton therapy facility and propose mathematical models that are capable of dealing with stochasticity. Thirdly, we deal with a different variant of well-known parallel machine scheduling problem that seeks to optimize nonidentical project assignments to dissimilar machines. We develop several exact solution methods for this complex problem in both mathematical programming and constraint programming contexts.

Recently, proton therapy has emerged as a new and fast growing form of radiation therapy for cancer patients. Currently, the demand for this form of treatment is increasing at a very high speed due to the effectiveness of the proton therapy even though the available capacity is highly limited (14 in use and 10 are under construction in the U.S.), mostly because it is a very expensive treatment procedure to deliver. Goitein and Jermann (2003) demonstrated that the cost-per-fraction in proton therapy is more than two times the cost-per-fraction of X-ray therapy. As a consequence, decision makers seek to improve the utilization of the resources to successfully deliver this treatment to as many people as they can. In fact, in radiation therapy, a treatment protocol consists not only of a prescribed total delivered dose to so-called targets, but must also specify treatment times, sequences, and frequencies (fractionation schedule). It is known that dose fractionation contributes to the preservation of healthy tissue throughout the treatment (see, e.g.,Yamada et al. (1999)) and a shorter fractionation schedule provides a more economical use of

the radiation therapy facilities while still improving, albeit marginally, a patient's quality of life (see, e.g., Shelley et al. (2000)). Under these factors, Chapters 2 and 3 of this research are dedicated to investigate the optimal strategies of allocating resources to the patients and quantify the trade-offs between different objectives in proton therapy facilities.

In Chapter 2, we introduce a patient scheduling problem which seeks to maximize the total number of fractions and total deviation from the patient mix restrictions in a planning horizon within the facility. Patient mix restrictions are specified by the managers of the facility in order to simplify and ease the resource allocation to the patients. Therefore, patients are categorized into several groups based on the similarities and differences of the treatment procedures offered in the facility. We study the optimal solution structure and derive analytical expressions for the single and multiple criteria models with different side constraints involved. Several practical rule of thumbs are derived to assess the impacts of the amount of available resources and patient mix restrictions. Our work on generating efficient frontiers for the bicriteria patient scheduling problem can be generalized to other planning and scheduling problems with any kind of entity mix restrictions.

Patient scheduling problem in proton therapy facilities with stochastic patient arrivals is studied in Chapter 3. We propose a Markov Decision Process (MDP) model to capture the optimal decision policies when the facility is operating at different states. The MDP model focuses on finding the best actions to be taken when the state of the facility (available capacity and the current number of patients from each group) is known. It provides the best immediate actions under uncertain demand. Due to the curse of dimensionality embedded in this multi-category model formulation, another MDP model is created through state aggregation technique. We show that the aggregate MDP model provides good approximate optimal patient admission policies and it takes significantly less computational time to tackle larger patient scheduling problem instances.

Chapter 4 introduces a variant of parallel machine scheduling problem that is commonly seen in supply chain scheduling. In this problem, projects and machines are assumed to be non-identical and dissimilar, respectively. Furthermore, projects may have several time restrictions

and machines may require sequence dependent setup times after they finish processing a project and before starting on the next one. This problem is modeled both in inter programming (IP) and constraint programming (CP) contexts. Real life problem instances prepared in collaboration with the U.S. Army Corps of Engineers (USACE) are used during our experiments. Logic-based Benders decomposition algorithms are proposed in order to solve the problem to optimality and measure the quality of the solutions obtained by IP and CP models.

Bibliography

Goitein, M. and Jermann, M. (2003). The relative costs of proton and x-ray radiation therapy. *Clinical Oncology*, 15:S37–S50.

Shelley, W., Brundage, M., Hayter, C., Paszat, L., Zhou, S., and Mackillop, W. (2000). A shorter fractionation schedule for postlumpectomy breast cancer patients. *International Journal of Radiation Oncology, Biology, Physics*, 47:1219–1228.

Yamada, Y., Ackerman, I., Franssen, E., Mackenzie, R. G., and Thomas, G. (1999). Does the dose fractionation schedule influence local control of adjuvant radiotherapy for early stage breast cancer. *International Journal of Radiation Oncology, Biology, Physics*, 44:99–104.

2. STRATEGIC LEVEL PATIENT SCHEDULING MATHEMATICAL MODELING IN A PROTON THERAPY FACILITY

Ridvan Gedik Chase Rainwater Edwin Romeijn

2.1 Introduction

Maximizing utilization of resources in proton therapy facilities has become one of the first priorities of healthcare planners in order to meet exponentially growing demand for this technology. It is a challenging task since several operational restrictions and their possible impacts on the facility capacity must be taken into account simultaneously. In this study, we aim to provide efficient solutions for capacity planning problems in a proton therapy facility and investigate the impact of various limitations as highlighted in Gedik (2011). These are (i) strategic patient mix constraints, (ii) physician availability, (iii) operating hours, (iv) availability of treatment gantries, (v) gantry specialization and (vi) gantry switching flexibility.

Patient mix optimization, requirement that the mix of patients treated satisfy desired percentages, is a relatively new consideration in proton therapy planning. These percentages are determined by the decision managers before each planning horizon in order to be used for the upcoming resource allocation problems. *Diagnosis Related Groups (DRGs)* is a very similar concept that has been used by health care providers to classify patient groups based on patient types (i.e. case mix, patient mix) and their treatment costs incurred by hospitals (Averill et al., 1998). DRGs play an important role in Medicare's hospital reimbursement system, whereas a hospital's case mix or patient mix preferences determine the costs of all the services provided for different patient types. From a managerial perspective, case mix preferences can address how to plan resources with respect to the needs of different patient types and, from a clinical perspective, they refer to the conditions of patients that are being treated in the hospital and the difficulty level of associated treatments (Averill et al., 1998). Hence, a well planned patient mix preferences in a facility can improve the efficiency of resources (i.e. physicians, machines) and help health care providers position their treatment capabilities with the projected treatment types required by pa-

tients.

Gedik (2011) propose a bicriteria mathematical programming model that determines the best patient admission policy for a proton therapy facility maximizing the number of treatment sessions and minimizing the deviations from the patient mix preferences over a finite planning horizon. In order to be able to assess the tradeoffs between these two objectives, they generate efficient frontiers by utilizing a technique entitled *Noninferior Set Estimation (NISE)* method developed by Cohon et al. (1979). In this chapter, we study the mathematical structures of this patient scheduling problem with and without strict patient mix requirements and provide optimal solutions on the best patient admission policies under several side constraints. Moreover, we develop a polynomial time exact solution algorithm which is capable of generating the exact efficient frontier for so called *base* patient scheduling problem in Gedik (2011). Our solution techniques are applicable to all healthcare treatment facilities as this study is motivated by the collaboration with the University of Florida Proton Therapy Institute (UFPTI) in Jacksonville, Florida.

The remainder of this chapter is organized as follows. Section 2.2 summarizes previous efforts on solving patient scheduling problems. Section 2.3 introduces the proton therapy patient scheduling problem in Gedik (2011). Solution structures and analytical derivation of optimal solutions are discussed in Section 2.4 and 2.5. Finally, Section 2.5.6 demonstrates how well the proposed algorithm performs over the NISE method.

2.2 Literature Review

Healthcare scheduling with a particular focus on operating & emergency room, nurse, physician and patient planning has been widely studied over the past 50 years. A very detailed review on operating room planning is proposed by Cardoen et al. (2010) which groups previous work in terms of mathematical structures and technical features (i.e. patient type & demand, appointment system). Patients are described in two different groups: (i) elective and (ii) non-elective. Elective patient is the one whose operation is planned in advance, whereas non-elective patient

requires unexpected operation. Inpatient (required overnight stay) and outpatient (discharge on same day) subgroups are listed under elective patient group as well. In order to reduce uncertainties associated with patient attributes (i.e., patient type, financial gain, resource allocation), many researchers focus on healthcare planning problems with elective patients (e.g. Bowers and Mould (2005), Cayirli et al. (2006), Pham and Klinkert (2008), Conforti et al. (2008)).

Based on the patient type classification in Cardoen et al. (2010), patients in our study belong to elective outpatient group in comparison to elective inpatient group that is used in several patient mix optimization problems (see, e.g. Adan and Vissers (2002), Vissers et al. (2005), Adan et al. (2009)). This is in parallel with the reality because consecutive daily treatment sessions are prescribed for proton therapy patients and they do not typically spend the night in the facility. Length of stay and consecutive daily treatment sessions are the two important factors to be considered in allocating resources in a healthcare facility for inpatient and outpatient groups, respectively.

Maximizing the number of treated patients (see, e.g., Conforti et al. (2010), Ballard and Kuhl (2006), and Cardoen and Demeulemeester (2008)) and minimizing the waiting times of patients (see, e.g., Chaabane et al. (2008) and Kaandorp and Koole (2007)) are the two most desired objectives in patient scheduling problems subject to several different constraints. Cardoen et al. (2010) suggest that these two objectives complement each other since the average number of patients in the system can be approximately calculated by multiplying the average cycle time (treatment time and waiting time) by the average throughput of the system (Little's Law). Hence, Bosch and Dietz (2000) state that reduced waiting times for the patients implies more capacity/resource availability and ultimately, an increased number of (new) treated patients. For instance, patient scheduling problem in a radiation therapy clinic is formulated as an integer linear optimization problem by Conforti et al. (2010) in which the objective is to maximize the number of new scheduled patients. Weights are determined for each patient group based on the pathological conditions that are later used to assess the priority levels for the treatments in waiting lists.

During the early stages of DRGs implementations, linear programming (LP) mathemati-

cal models are developed to find the optimal case mix in a hospital by utilizing the well known *product mix* problem formulation (Hughes and Soliman (1985), Robbins and Tuntiwongpiboon (1989)). In these LPs, the objective is to maximize the total profit margins by treating patients from different DRGs subject to resource availability and demand satisfaction constraints. It is later shown that case mix preferences play a crucial role in developing a strategic long term plan for a hospital since these mix levels directly or indirectly impacts type and amount of resources, income and cost preferences required for a sustainable health care service (Blake and Carter, 2002). Vanberkel et al. (2011) provide a stochastic mathematical model that seeks to obtain the best patient mix levels, which patient types to treat in the hospital, in order to achieve the maximum benefit. In this study, patient types are considered as projects as in *project sequencing problem* and an approximate solution approach is developed to determine the sequence in which the patient types should be accepted. Mulholland et al. (2005) formulate an LP model to identify the the optimal mix of surgical procedures. With the help of this model, they conduct sensitivity analysis to measure the impacts of different surgery mix levels on financial outcomes. They observe significant financial improvements by just allowing no more than 15% deviation in procedure mix without any capacity investments.

The papers discussed above have long term strategic planning periods and most of them aim to find the optimal case/patient/surgery mix preferences subject to generic capacity & demand constraints. Tactical level patient mix optimization problems are also studied by researchers (e.g. Adan et al. (2009), Vissers et al. (2005)). However, these problems either account for a shorter planning period or consider only a specific unit of the facility.

Patient scheduling problem in this study and other healthcare applications have similar aspects such as treatment continuity, capacity, staff/physician requirements etc. One of the differences is that all patient groups are considered to have same urgency levels in our model. Therefore, we do not assign priority weights to prioritize patients or treatment types. Patients are only categorized to account for different treatment times and number of fractions to be delivered for each category. This study also differs from others in the patient mix optimization literature. Pa-

tient mix levels in other studies are treated as outcomes of the models to maximize the benefits, whereas they are treated as inputs in this dissertation and their effects in overall capacity utilization are quantified in terms of number of treatment sessions.

In healthcare scheduling problems, uncertainty associated with demand is one of the major challenges needs to be tackled in order to produce efficient and robust solutions. Canceled appointments and no-shows are the two other most common sources of uncertainty in patient scheduling. Due to the long waiting lists for the proton therapy, we assume that the replacement of any canceled appointment or no-show with a patient on the waiting list can be made instantaneously. This enables us to assume deterministic patient arrivals to the facility while building our models in this chapter. However, this assumption is relaxed in Chapter 2.

The encouraging findings in the previous patient scheduling studies have motivated further efforts in developing advanced mathematical models for efficient patient scheduling in proton therapy facilities. Therefore, one of our primary objectives is to illustrate the effects of a given patient mix restrictions on the capacity of a proton therapy facility. Gedik (2011) point out that it is very difficult to satisfy the patient mix restrictions precisely. Hence, they propose a bicriteria mathematical model that seeks to maximize the total treatment sessions and minimize the total deviation from the desired patient mix levels. Our efforts in this study demonstrate the optimal solutions for this problem under different scenarios.

2.3 Mathematical Models for the Proton Therapy Patient Scheduling

This section introduces the mathematical models proposed by first Men (2009) and then Gedik (2011) to solve the patient scheduling problem in proton therapy facilities. The notation (see Table 4.1) used in Gedik (2011) is adopted throughout the remainder of this chapter. The baseline (base) model and all side constraints related to varying operational restrictions are presented in this section for the reader's reference.

Table 2.1: Notation

	Sets
T	set of days in planning horizon
G	set of gantries
K	set of patient categories
$K^a \in K$	set of patient categories needing anesthesia for treatment
$K^2 \in K$	set of patient categories needing twice-a-day fractions
$T_{NA} \in T$	set of non authorized days to start new patients
$T_A \in T$	set of authorized days to start new patients
	Parameters
C_{tg}	time available for treatment on gantry g on day t
n_k	number of consecutive treatment days for patients in category k
f_k	number of fractions required on each day of the treatment by a patient in category k
c_k	duration of a fraction on each day of the treatment by a patient in category k
$\bar{c}_k$	setup time for the first fraction on the first day for patients in category k
d_k	desired fraction of patients in category k treated over the planning horizon
A_g	anesthesia team availability per day on gantry g
γ	minimal time between two fractions (in minutes)
η_g	Available prime hours on authorized days to treat new patients
	Decision variables
x_{tkg}	number of new patients in category k that start their treatment on day t on gantry g
y_{tkg}	number of patients in category k that receive treatment on day t on gantry g
u_k	deviation from target level d_k for patient category $k \in K$

2.3.1 Base Model

Three fundamental restrictions in a proton therapy facility constitute the base model. These restrictions are (i) gantry capacity limitations, (ii) patient continuity requirements and (iii) patient mix specifications. Since one of the objectives of this study is to demonstrate a strategic capacity analysis of the proton therapy facility, we would like to prevent end of study effects by assuming that the facility is operating in steady-state. Hence, while modeling an integer programming model, we consider the planning period to be cyclic with a period length of T days. Another work around for the end of study effects is to assume an infinite horizon length which apparently yields in infinite number of constraints and variables when the same integer program is employed. In Section 2.4.1.1, we discuss how finite horizon model (B) can be reformulated as an infinite horizon model without having infinite number of decision variables and constraints.

$$\text{maximize } \frac{1}{T}\sum_{t=1}^{T}\sum_{k=1}^{K}\sum_{g=1}^{G} f_k y_{tkg}$$

subject to (B)

$$\sum_{n=1}^{n_k} x_{[t-n+1],kg} = y_{tkg} \qquad t \in T; k \in K; g \in G \tag{2.1}$$

$$\sum_{k=1}^{K} \bar{c}_k x_{tkg} + \sum_{k=1}^{K} c_k f_k y_{tkg} \leq C_{tg} \qquad t \in T; g \in G \tag{2.2}$$

$$\sum_{t=1}^{T}\sum_{g\in G} x_{tkg} = d_k \sum_{t=1}^{T}\sum_{k'=1}^{K}\sum_{g\in G} x_{tk'g} \qquad k \in K \tag{2.3}$$

$$x_{tkg}, y_{tkg} \geq 0 \qquad t \in T; k \in K; g \in G$$

The objective function seeks to maximize the average number of fractions treated across the planning period, while attempting to adhere to a specified mixture of patient categories treated (2.3). Treatment continuity and capacity limitations are enforced by constraints (2.1) and (2.2), respectively. Constraints (2.1) assure that appropriate relationship between two decision variables our problem exists; number of fractions and number of patients. By this way, delivering fractions over a number of consecutive (week)days is guaranteed.

2.3.2 Bicriteria Base Model

Constraints (2.3) restrict total number of patients treated from type k to be equivalent to multiplication of desired percentage d_k of patient type k and total number of patients accepted to the facility across all patient categories. Being able to monitor the behavior of optimal total fraction level as patient types deviate from their desired percentage levels provides significant information for decision makers in deciding how to allocate facility resources (i.e. gantry hours, anesthesia teams etc.) in case of several other operational side constraints. As a consequence, we propose a

bicriteria base model (*BB*) that seeks to maximize the average number of total fractions delivered and minimize the total deviations from patient mix level.

$$\text{maximize} \left\{ \frac{1}{T} \sum_{t=1}^{T} \sum_{k=1}^{K} \sum_{g=1}^{G} f_k y_{tkg}, -\sum_{k=1}^{K} u_k \right\}$$

subject to (BB)

(2.1)

(2.2)

$$\sum_{t=1}^{T} \sum_{g \in G} x_{tkg} \geq d_k \sum_{t=1}^{T} \sum_{k'=1}^{K} \sum_{g \in G} x_{tk'g} - u_k \qquad k \in K \qquad (2.4)$$

$$\sum_{t=1}^{T} \sum_{g \in G} x_{tkg} \leq d_k \sum_{t=1}^{T} \sum_{k'=1}^{K} \sum_{g \in G} x_{tk'g} + u_k \qquad k \in K \qquad (2.5)$$

$$u_k \geq 0 \qquad k \in K$$

$$x_{tkg}, y_{tkg} \geq 0 \qquad t \in T; k \in K; g \in G$$

Strict patient mix constraints are partitioned into two different sets of constraints. Constraints (2.4) and (2.5) allow the model to treat patients below and above their target levels that are quantified by introducing u_k (deviation level for category k) decision variables.

2.3.3 Side Constraints

This section shows how the operational restrictions are modeled in Gedik (2011) in addition to the ones already introduced in (B) and (BB).

2.3.3.1 Anesthesia Patients

Constraints (2.6) make sure that the treatment durations for the patients who need anesthesia surveillance during the sessions do not exceed the daily availability of anesthesia teams in each

gantry.

$$\sum_{k \in K^a} (\bar{c}_k x_{tkg} + c_k f_k y_{tkg}) \leq A_g \qquad t \in T; g \in G \tag{2.6}$$

2.3.3.2 BID Patients

Some patient groups may need to receive two treatment sessions on the same day. They are called BID (*Bis In Die*, twice daily) patients and constraints (2.7) assure that their second treatment session starts at least $\gamma + \max_{k \in K^2} c_k$ time units after the start of the day.

$$\sum_{k \in K^2} \left(\bar{c}_k x_{tkg} + c_k y_{tkg}\right) \leq C_{tg} - \tau - \max_{k \in K^2} c_k \qquad t \in T; g \in G \tag{2.7}$$

2.3.3.3 Gantry Specialization

Healthcare planners may specialize the use of some gantries for only certain patient groups in order to increase the service level. The following constraints are used if the patient categories in the set K_g are specialized to be treated in gantry $g \in G$.

$$x_{tkg} = 0 \qquad t \in T; g \in G;\ k \in K \backslash K_g \tag{2.8}$$

2.3.3.4 Prime Hours

Enforcing new patients to start their treatments only on the listed authorized days (T_A) during prime hours (η_g) in gantry g is handled by constraints (2.9).

$$\sum_{k \in K} (\bar{c}_k + c_k) x_{tkg} \leq \eta_g \qquad t \in T_A; g \in G \tag{2.9}$$

$$x_{tkg} = 0 \qquad t \in T_{NA}; g \in G;\ k \in K \tag{2.10}$$

Hence, starting patients on non authorized days (T_{NA}) is ensured by constraints (2.10).

2.4 Characterizing Optimal Solutions to (B) and (BB)

2.4.1 Optimal Solution Structure

In this section we show that, for certain model variants, the model (B) and (BB) have time invariant (stationary) optimal solutions. Notation below is used to represent different daily fraction requirement and aggregated reward for treating a single patient of each patient type for both problem (B) and (BB).

- r_k = reward for treating a patient in category k (often $rk = n_k f_k$)
- c_{kn} = total treatment time on day n of treatment for patients in category k ($n = 1,\ldots,n_k$, $k = 1,\ldots,K$). Note that in our earlier notation, $c_{k1} = \bar{c}_k + f_k c_k$ and $c_{kn} = f_k c_k$ for $n = 2,\ldots,nk$.

2.4.1.1 Base Model

$$\text{maximize } \frac{1}{T}\sum_{t=1}^{T}\sum_{k=1}^{K}\sum_{g=1}^{G} r_k x_{tkg}$$

subject to (B)

$$\sum_{k=1}^{K}\sum_{n=1}^{n_k} c_{kn} x_{[t-n+1],kg} \leq C_{tg} \qquad t \in T;\, g \in G \tag{2.11}$$

$$\sum_{t=1}^{T}\sum_{g\in G} x_{tkg} = d_k \sum_{t=1}^{T}\sum_{k'=1}^{K}\sum_{g\in G} x_{tk'g} \qquad k \in K \tag{2.12}$$

$$x_{tkg} \geq 0 \qquad t \in T;\, k \in K;\, g \in G$$

The following lemma shows that, when gantry capacities are stationary, this problem has a stationary optimal solution.

Lemma 2.4.1 *Suppose that the gantry capacities are time-invariant, i.e., $C_{tg} = C_g$ for all $t = 1,\ldots,T$ and $g = 1,\ldots,G$. Then there exists a stationary feasible solution to this problem.*

Proof Suppose x is a feasible solution to (B). Then define the following alternative solution:

$$x'_{tkg} = \frac{1}{T}\sum_{t'=1}^{T} x_{t'kg} \qquad\qquad t \in T; k \in K; g \in G.$$

We first show that x' is a feasible solution to (B):

- Gantry capacity constraints:

$$\begin{aligned}
\sum_{k=1}^{K}\sum_{n=1}^{n_k} c_{kn} x'_{[t-n+1],kg} &= \sum_{k=1}^{K}\sum_{n=1}^{n_k} c_{kn}\frac{1}{T}\sum_{t'=1}^{T} x_{[t'-n+1],kg} \\
&= \frac{1}{T}\sum_{t'=1}^{T}\sum_{k=1}^{K}\sum_{n=1}^{n_k} c_{kn} x_{[t'-n+1],kg} \\
&\leq \frac{1}{T}\sum_{t'=1}^{T} C_{t'g} \\
&= C_g \\
& t \in T; g \in G
\end{aligned}$$

- Patient mix constraints:

$$\begin{aligned}
\sum_{g\in G} x'_{tkg} &= \sum_{g\in G}\frac{1}{T}\sum_{t'=1}^{T} x_{t'kg} \\
&= \frac{1}{T}\sum_{t'=1}^{T}\sum_{g\in G} x_{t'kg} \\
&= d_k \frac{1}{T}\sum_{t'=1}^{T}\sum_{k'=1}^{K}\sum_{g\in G} x_{t'k'g} \\
&= d_k \sum_{k'=1}^{K}\sum_{g\in G}\frac{1}{T}\sum_{t'=1}^{T} x_{t'k'g} = \\
&= d_k \sum_{k'=1}^{K}\sum_{g\in G} x'_{tk'g} \qquad\qquad t \in T\ ; k \in K
\end{aligned}$$

  which actually shows that the patient mix constraints are satisfied in each period, and therefore also on average with respect to $t \in T$.

- Nonnegativity constraints: Due to the nonnegativity of $x_{tkg}\ \forall\, t \in T; k \in K; g \in G$, we can

write

$$x'_{tkg} = \frac{1}{T}\sum_{t'=1}^{T} x_{t'kg} \geq 0 \qquad\qquad t \in T; k \in K; g \in G.$$

- Objective function: Since x' is stationary, for any t, we can write

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T}\sum_{k=1}^{K}\sum_{g=1}^{G} r_k x'_{tkg} &= \sum_{k=1}^{K}\sum_{g=1}^{G} r_k x'_{tkg}. \\ &= \sum_{k=1}^{K}\sum_{g=1}^{G} r_k \frac{1}{T}\sum_{t=1}^{T} x_{t'kg} \\ &= \frac{1}{T}\sum_{t=1}^{T}\sum_{k=1}^{K}\sum_{g=1}^{G} r_k x_{tkg}.\end{aligned}$$

  Finally, we show that x' has the same objective function value as x.

□

The steps above imply that the problem can be simplified considerably by letting

$$x_{tkg} = \bar{x}_{kg} \qquad\qquad t = 1, \ldots, \infty; k \in K; g \in G.$$

Then the problem with stationary gantry capacities can now be written as

$$\text{maximize} \sum_{k=1}^{K}\sum_{g=1}^{G} r_k \bar{x}_{kg}$$

subject to (𝕀)

$$\sum_{k=1}^{K}\sum_{n=1}^{n_k} c_{kn}\bar{x}_{kg} \leq C_g \qquad\qquad g \in G \tag{2.13}$$

$$d_k \sum_{k'=1}^{K}\sum_{g=1}^{G} \bar{x}_{k'g} = \sum_{g=1}^{G} \bar{x}_{kg} \qquad\qquad k \in K \tag{2.14}$$

$$\bar{x}_{kg} \geq 0 \qquad\qquad k \in K; g \in G.$$

Note that problem 𝕀 is independent of the time horizon T, so that its optimal solution solves problem (B) with any horizon, and therefore an infinite-horizon variant of the problem.

2.4.1.2 Bicriteria Base Model

$$\text{maximize} \left\{ \frac{1}{T} \sum_{t=1}^{T} \sum_{k=1}^{K} \sum_{g=1}^{G} r_k x_{tkg}, -\sum_{k=1}^{K} u_k \right\}$$

subject to (BB)

$$\begin{aligned}
\sum_{k=1}^{K} \sum_{n=1}^{n_k} c_{kn} x_{[t-n+1],kg} &\leq C_{tg} && t \in T; g \in G \\
\left| \frac{1}{T} \sum_{t=1}^{T} \sum_{g=1}^{G} \left(d_k \sum_{k'=1}^{K} x_{tk'g} - x_{tkg} \right) \right| &\leq u_k && k \in K \\
x_{tkg} &\geq 0 && t \in T; k \in K; g \in G \\
u_k &\geq 0 && k \in K
\end{aligned}$$

The following lemma shows that, when gantry capacities are stationary, we can restrict model (*BB*) to stationary solutions.

Lemma 2.4.2 *Suppose that the gantry capacities are time-invariant, i.e., $C_{tg} = C_g$ for all $t = 1, \ldots, T$ and $g = 1, \ldots, G$. Then for any feasible solution to (BB), there exists a stationary feasible solution with the same objective function values.*

Proof Suppose (x, u) is a feasible solution to (*BB*). Then define the following alternative solution:

$$\begin{aligned}
x'_{tkg} &= \frac{1}{T} \sum_{t'=1}^{T} x_{t'kg} && t \in T; k \in K; g \in G \\
u' &= u.
\end{aligned}$$

We already know that (x', u') satisfies the gantry capacity and nonnegativity constraints. Furthermore, it has identical value for both objectives. It remains to show that this solution satisfies the relaxed patient mix constraints. To this end, first note that

$$\frac{1}{T} \sum_{t=1}^{T} \sum_{g=1}^{G} x_{tkg} = \sum_{g=1}^{G} \frac{1}{T} \sum_{t=1}^{T} x_{tkg} = \sum_{g=1}^{G} x'_{tkg} \qquad t \in T.$$

This implies that

$$\begin{aligned}
&\left|\frac{1}{T}\sum_{t=1}^{T}\sum_{g=1}^{G}\left(d_k\sum_{k'=1}^{K}x_{tk'g}-x_{tkg}\right)\right| \\
&=\left|d_k\sum_{k'=1}^{K}\frac{1}{T}\sum_{t=1}^{T}\sum_{g=1}^{G}x_{tk'g}-\frac{1}{T}\sum_{t=1}^{T}\sum_{g=1}^{G}x_{tkg}\right| \\
&=\left|d_k\sum_{k'=1}^{K}\sum_{g=1}^{G}x'_{t'k'g}-\sum_{g=1}^{G}x'_{t'kg}\right| \\
&=\left|\frac{1}{T}\sum_{t'=1}^{T}\sum_{g=1}^{G}\left(d_k\sum_{k'=1}^{K}x_{t'k'g}-x_{t'kg}\right)\right|\leq u_k \qquad\qquad t\in T\,;k\in K.
\end{aligned}$$

Therefore, the relaxed patient mix constraints, with $u'_k = u_k$ for $k = 1,\ldots,K$, are indeed satisfied.

□

Similar to the single criteria base model problem, we can reduce the finite horizon bicriteria problem (*BB*) to infinite horizon bicriteria problem by simply letting

$$\bar{x}_{kg} = x_{tkg} \qquad\qquad t = 1,\ldots,\infty;\, k\in K;\, g\in G.$$

Then problem (*BB*) with stationary gantry capacities can now be written as

$$\text{maximize}\left\{\sum_{k=1}^{K}\sum_{g=1}^{G}r_k\bar{x}_{kg},-\sum_{k=1}^{K}u_k\right\}$$

subject to (B-LP)

$$\sum_{k=1}^{K}\sum_{n=1}^{n_k}c_{kn}\bar{x}_{kg}\leq C_g \qquad g\in G \tag{2.15}$$

$$\left|\sum_{g=1}^{G}\left(d_k\sum_{k'=1}^{K}\bar{x}_{k'g}-\bar{x}_{kg}\right)\right|\leq u_k \qquad k\in K \tag{2.16}$$

$$\bar{x}_{kg}\geq 0 \qquad k\in K;\, g\in G$$

$$u_k\geq 0 \qquad k\in K.$$

Problem (B-LP) is also independent of the time horizon T and its optimal solution solves problem (*BB*) with any horizon. Hence, (B-LP) is an infinite-horizon variant of problem (*BB*).

2.4.1.3 Side Constraints

This section demonstrates that model (B) and (BB) preserve the time-invariant optimal solutions with the following side constraints.

- Gantry switching:

  When gantry switching is allowed, the gantry capacity constraints are replaced by a single constraint in which both left and right hand side aggregated over all gantries as follows:

  $$\sum_{g=1}^{G}\sum_{k=1}^{K}\sum_{n=1}^{n_k} c_{kn}x_{[t-n+1],kg} \leq \sum_{g=1}^{G} C_{tg} \qquad\qquad t \in T.$$

  This implies that when gantry switching is allowed and the necessary condition ($C_{tg} = C_g$) for Lemma 2.4.1 and Lemma 2.4.2 are held, x' and (x', u') are feasible solutions to (B) and (BB), respectively.

- Anesthesia patients:

  Let $A_{tg} = A_g$ for each $t \in T$. We assume that the set of anesthesia patients (K^a) is stationary. Then, we can write

  $$\begin{aligned}
  \sum_{k\in K^a}\sum_{n=1}^{n_k} c_{kn}x_{tkg} &= \sum_{k\in K^a}\sum_{n=1}^{n_k} c_{kn}\frac{1}{T}\sum_{t'=1}^{T} x_{t'kg} \\
  &= \frac{1}{T}\sum_{t'=1}^{T}\sum_{k\in K^a}\sum_{n=1}^{n_k} c_{kn}x_{t'kg} \\
  &\leq \frac{1}{T}\sum_{t'=1}^{T} A_{t'g} \\
  &= A_g \qquad\qquad t \in T;\ g \in G.
  \end{aligned}$$

  Therefore, when the anesthesia team availability is time invariant, x' and (x', u') are feasible solutions to (B) and (BB), respectively, with anesthesia patients.

- BID patients:

  Let us define $B_{tg} = C_{tg} - \gamma - \max_{k\in K^2} c_k$. If the set of BID patients (K^2) is stationary and

assuming $B_{tg} = B_g$ for each $t \in T$, we can write

$$\begin{aligned}
\sum_{k\in K^2}\sum_{n=1}^{n_k} c_{kn}x_{tkg} &= \sum_{k\in K^2}\sum_{n=1}^{n_k} c_{kn}\frac{1}{T}\sum_{t'=1}^{T} x_{t'kg} \\
&= \frac{1}{T}\sum_{t'=1}^{T}\sum_{k\in K^2}\sum_{n=1}^{n_k} c_{kn}x_{t'kg} \\
&\le \frac{1}{T}\sum_{t'=1}^{T} B_{t'g} \\
&= B_g \qquad\qquad t \in T;\ g \in G.
\end{aligned}$$

Thus, x' and (x', u') are feasible solutions to (B) and (BB), respectively, with BID patients.

- Gantry specialization:

  We assume that the specialized patient categories in the set K^g for gantry g is stationary. Then, x' and (x', u') are feasible solutions to (B) and (BB), respectively, with gantry specialization restriction.

$$\begin{aligned}
x_{tkg} &= 0 \qquad && t \in T;\ g \in G;\ k \in K;\ k \notin K^g \\
\frac{1}{T}\sum_{t'=1}^{T} x_{t'kg} &= 0 \qquad && t \in T;\ g \in G;\ k \in K;\ k \notin K^g \\
x_{t'kg} &= 0 \qquad && t \in T;\ g \in G;\ k \in K;\ k \notin K^g
\end{aligned}$$

- Cyclic capacities:

  Suppose gantry capacities are not stationary but cyclic with cycle length T_0, and suppose that $T = aT_0$ for some integer a. That is, $C_{(l-1)T_0+\tau,g} = C_{\tau g}$ for $g = 1,\ldots,G$, $l = 1,\ldots,a$, $\tau = 1,\ldots,T_0$. For any given feasible solution to (B), we now define the following cyclic solution and follow a similar analysis as above:

$$x'_{(l-1)T_0+\tau,kg} = \frac{1}{a}\sum_{l'=1}^{a} x_{(l'-1)T_0+\tau,kg} \quad \tau = 1,\ldots,T_0;\ l = 1,\ldots,a;\ k = 1,\ldots,K;\ g = 1\ldots,G.$$

We obtain an LP over the cycle length T_0 by defining

$$\bar{x}_{\tau kg} = x_{(l'-1)T_0+\tau,kg} \qquad \tau = 1,\ldots,T_0;\ l = 1,\ldots,a;\ k = 1,\ldots,K;\ g = 1\ldots,G.$$

The base model then becomes

$$\text{maximize } \frac{1}{T_0}\sum_{\tau=1}^{T_0}\sum_{k=1}^{K}\sum_{g=1}^{G} r_k \bar{x}_{\tau kg}$$

subject to

$$\sum_{k=1}^{K}\sum_{n=1}^{n_k} c_{kn}\bar{x}_{[\tau-n+1],kg} \leq C_{\tau g} \qquad \tau = 1,\ldots,T_0;\ g \in G$$

$$\sum_{\tau=1}^{T_0}\sum_{g\in G}\bar{x}_{\tau kg} = d_k \sum_{\tau=1}^{T_0}\sum_{k'=1}^{K}\sum_{g\in G}\bar{x}_{\tau k'g} \qquad k \in K$$

$$\bar{x}_{\tau kg} \geq 0 \qquad \tau = 1,\ldots,T_0;\ k \in K;\ g \in G$$

where the [.] notation now is relative to T_0 rather than T. Note that all conclusions obtained through Lemma 2.4.1 for the base model and extensions on other side constraints remain valid with the cyclic capacities as described above. For example, if T_A in prime hours side constraints follows the described cyclic pattern, then there exists a time invariant stationary solution to (B) with prime hours constraints.

2.4.2 Deriving Optimal Solution Values

The results of the previous section suggest a specific structure to the optimal solution of (B) and (BB). In this section, we make use of these results to analytically obtain the optimal solution (B) with other model variants.

2.4.3 Base Model

Recall that optimal solution to (B) is stationary which leads to more compact representation ($\mathbb{I}$) of our problem. Therefore, we will focus on obtaining an optimal solution to $\mathbb{I}$. Note that the objective function of problem ($\mathbb{I}$) can be rewritten as

$$\begin{aligned}\sum_{k=1}^{K}\sum_{g=1}^{G} r_k \bar{x}_{kg} &= \sum_{k=1}^{K} r_k \sum_{g=1}^{G} \bar{x}_{kg} \\ &= \sum_{k=1}^{K} r_k d_k \sum_{k'=1}^{K}\sum_{g=1}^{G} \bar{x}_{k'g} \\ &= \tilde{r} \sum_{k'=1}^{K}\sum_{g=1}^{G} \bar{x}_{k'g}\end{aligned}$$

where $\tilde{r} = \sum_{k=1}^{K} r_k d_k$. This implies that *maximizing the total reward is equivalent to maximizing the number of patients treated* through optimization problem ($\mathbb{I}$). Let $g' \in G'$ be the set of gantries in which there are some slack capacities left ($G' \in G$).

$$\sum_{k=1}^{K}\sum_{n=1}^{n_k} c_{kn}\bar{x}_{kg} < C_{g'} \qquad\qquad g' \in G'$$

For each $g' \in G'$, modify the solution $\bar{x}$ to get $\bar{x}'$ by using v_k values where $v_k \geq 0$ as follows:

$$\begin{aligned}\bar{x}'_{kg} &= \bar{x}_{kg} + v_k && k \in K \\ d_k \sum_{k'=1}^{K}\sum_{g'=1}^{G} \bar{x}'_{k'g'} &= \sum_{g'=1}^{G} \bar{x}'_{kg'} && k \in K \\ v_k &\geq 0 && k \in K\end{aligned}$$

The patient mix constraints become

$$\begin{aligned}d_k \sum_{k'=1}^{K}\left(\sum_{g'=1}^{G} \bar{x}'_{k'g'} + v_{k'}\right) &= \sum_{g'=1}^{G} \bar{x}'_{kg'} + v_{k'} && k \in K \\ d_k \sum_{k'=1}^{K} v_{k'} &= v_k && k \in K \qquad (2.17)\end{aligned}$$

We can choose a v such that the capacity constraint of gantry g' becomes binding.

$$v_k = v \tag{2.18}$$

$$v_{k'} = \frac{d_{k'}}{d_k} v \qquad k' \in K;\ k' \neq k. \tag{2.19}$$

We can tighten all nonbinding gantry capacity constraints by repeating these steps which explicitly increase the number of patients treated. Therefore, in the process of utilizing the unused gantry capacity, the objective function value increases. This implies that there exists an optimal solution in which all gantry capacity constraints are binding. Therefore, the optimization problem $\mathbb{I}$ reduces to

$$\text{maximize } \tilde{r} \sum_{k=1}^{K} \sum_{g=1}^{G} \bar{x}_{kg}$$

subject to

$$\sum_{k=1}^{K} \sum_{n=1}^{n_k} c_{kn} \bar{x}_{kg} = C_g \qquad g \in G$$

$$d_k \sum_{k'=1}^{K} \sum_{g=1}^{G} \bar{x}_{k'g} = \sum_{g=1}^{G} \bar{x}_{kg} \qquad k \in K$$

$$\bar{x}_{kg} \geq 0 \qquad k \in K;\ g \in G.$$

Together with gantry capacity and patient mix constraints we can write

$$\sum_{g=1}^{G} \sum_{k=1}^{K} \sum_{n=1}^{n_k} c_{kn} \bar{x}_{kg} = \sum_{g=1}^{G} C_g$$

$$\sum_{k=1}^{K} \sum_{n=1}^{n_k} c_{kn} \sum_{g=1}^{G} \bar{x}_{kg} = \sum_{g=1}^{G} C_g$$

$$\sum_{k=1}^{K} \sum_{n=1}^{n_k} c_{kn} d_k \sum_{k'=1}^{K} \sum_{g=1}^{G} \bar{x}_{k'g} = \sum_{g=1}^{G} C_g$$

$$\sum_{k'=1}^{K} \sum_{g=1}^{G} \bar{x}_{k'g} = \frac{\sum_{g=1}^{G} C_g}{\sum_{k=1}^{K} \sum_{n=1}^{n_k} c_{kn} d_k}.$$

Therefore, the objective function value of any optimal feasible solution for problem $\mathbb{I}$ is equal to

$$\sum_{g=1}^{G} C_g \frac{\sum_{k=1}^{K} r_k d_k}{\sum_{k=1}^{K}\sum_{n=1}^{n_k} c_{kn} d_k}. \tag{2.20}$$

Let $\bar{x}_{kg} = 0$ and $\bar{x}'_{kg} = v_k$. Therefore, we can choose v such that

$$\begin{aligned} v_k =& v = d_k \frac{C_g}{\sum_{k=1}^{K}\sum_{n=1}^{n_k} c_{kn} d_k} \\ v_{k'} =& \frac{d_{k'}}{d_k} v \\ =& \frac{d_{k'}}{d_k} d_k \frac{C_g}{\sum_{k=1}^{K}\sum_{n=1}^{n_k} c_{kn} d_k} \\ =& d_{k'} \frac{C_g}{\sum_{k=1}^{K}\sum_{n=1}^{n_k} c_{kn} d_k} \qquad\qquad k' \in K;\ k' \neq k \end{aligned}$$

or

$$\bar{x}_{kg} = \bar{x}'_{kg} = v_k = d_k \frac{C_g}{\sum_{k=1}^{K}\sum_{n=1}^{n_k} c_{kn} d_k} \qquad k \in K;\ g \in G. \tag{2.21}$$

This result for the base model can be extended to account for a variety of the operational side constraints discussed throughout this chapter.

- Gantry switching:

  In the case of stationary gantry capacities, we show that the gantry capacity constraint will be binding in the optimal solution. Thus, regardless of whether gantry switching is allowed or not, the optimal objective function and solution of problem ($\mathbb{I}$) are given in (2.20) and (2.21), respectively.

- Anesthesia patients:

  Anesthesia patient surveillance requirement independent of time horizon can be rewritten as below

$$\sum_{k \in K^a} \sum_{n=1}^{n_k} c_{kn} \bar{x}_{kg} \leq A_g \qquad\qquad g \in G.$$

In case of any slack either in gantry capacity or anesthesia constraint, we can modify feasible solution $\bar{x}$ to get $\bar{x}'$ by using v_k values given in (2.18) and (2.19). Since increasing $\bar{x}$ also increases the objective function value, there exists an optimal solution in which either or both of gantry capacity constraints and anesthesia requirement constraints are binding. Therefore, the optimization problem $\mathbb{I}$ with anesthesia patients restriction reduces to

$$\text{maximize } \tilde{r} \sum_{k=1}^{K} \sum_{g=1}^{G} \bar{x}_{kg}$$

subject to

$$\begin{aligned}
\sum_{k=1}^{K} \sum_{n=1}^{n_k} c_{kn} \sum_{g=1}^{G} \bar{x}_{kg} = \sum_{k=1}^{K} \sum_{n=1}^{n_k} c_{kn} d_k \sum_{k'=1}^{K} \sum_{g=1}^{G} \bar{x}_{k'g} \leq \sum_{g=1}^{G} C_g \\
\sum_{k\in K^a} \sum_{n=1}^{n_k} c_{kn} \sum_{g=1}^{G} \bar{x}_{kg} = \sum_{k\in K^a} \sum_{n=1}^{n_k} c_{kn} d_k \sum_{k'=1}^{K} \sum_{g=1}^{G} \bar{x}_{k'g} \leq \sum_{g=1}^{G} A_g \\
d_k \sum_{k'=1}^{K} \sum_{g=1}^{G} \bar{x}_{k'g} = \sum_{g=1}^{G} \bar{x}_{kg} \qquad k \in K \\
\bar{x}_{kg} \geq 0 \qquad k \in K; g \in G.
\end{aligned}$$

The optimal objective function value to the optimization problem above is shown in (2.22).

$$\sum_{k=1}^{K} r_k d_k \min \left\{ \frac{\sum_{g=1}^{G} C_g}{\sum_{k=1}^{K} \sum_{n=1}^{n_k} c_{kn} d_k}, \frac{\sum_{g=1}^{G} A_g}{\sum_{k\in K^a} \sum_{n=1}^{n_k} c_{kn} d_k} \right\} \tag{2.22}$$

- BID patients:

  The time-independent BID patient constraint can be written as follows

$$\sum_{k\in K^2} \sum_{n=1}^{n_k} c_{kn} \bar{x}_{kg} \leq C_g - \gamma - \max_{k\in K^2} c_k = B_g \qquad g \in G.$$

  Despite different patient sets (K^a and K^2), anesthesia and BID patients constraints have identical structure. Therefore, following same steps as in finding the optimal solution for problem $\mathbb{I}$ with anesthesia patients leads to the optimal objective function of problem $\mathbb{I}$ with

BID patients as seen in (2.23).

$$\sum_{k=1}^{K} r_k d_k \min\left\{ \frac{\sum_{g=1}^{G} C_g}{\sum_{k=1}^{K}\sum_{n=1}^{n_k} c_{kn} d_k}, \frac{\sum_{g=1}^{G} B_g}{\sum_{k\in K^2}\sum_{n=1}^{n_k} c_{kn} d_k} \right\} \tag{2.23}$$

2.5 Bicriteria Base Model (BB) Efficient Frontier

In previous sections, we dealt explicitly with the solution structure of various linear programming model variants. Specifically, Section 2.4.1.2 showed that any feasible solution to the bicriteria problem can be equivalently represented as a stationary solution. In this section, we seek to analytically characterize and produce the entire Pareto efficient frontier associated with (B-LP). Since, as we discuss in this section, the efficient frontier for our problem is a piecewise linear function in the total allowed patient mix deviation (see Figure 2.1), we give particular attention to determining the breakpoints associated with that function.

To begin the discussion, note that a point on the Pareto efficient frontier of problem (B-LP) can be obtained by solving the following problem with a given amount of allowed total deviation, U, where $\sum_{k=1}^{K} u_k = U$.

$$\text{maximize} \sum_{k=1}^{K}\sum_{g=1}^{G} r_k \bar{x}_{kg}$$

subject to (B-LP(U))

$$\sum_{k=1}^{K} w_k \bar{x}_{kg} \leq C_g \qquad g \in G \qquad [\alpha_g]$$

$$-u_k - \sum_{g \in G} \bar{x}_{kg} + d_k \sum_{k'=1}^{K} \sum_{g \in G} \bar{x}_{k'g} \leq 0 \qquad k \in K \qquad [\gamma_k]$$

$$-u_k + \sum_{g \in G} \bar{x}_{kg} - d_k \sum_{k'=1}^{K} \sum_{g \in G} \bar{x}_{k'g} \leq 0 \qquad k \in K \qquad [\mu_k]$$

$$\sum_{k=1}^{K} u_k = U \qquad [q]$$

$$\bar{x}_{kg} \geq 0 \qquad k \in K\ , g \in G$$

$$u_k \geq 0 \qquad k \in K.$$

To simplify notation, we've let $\sum_{n=1}^{n_k} c_{kn} = w_k$. (B-LP(U)) can be further simplified using the result of the following lemma.

Lemma 2.5.1 *Suppose that the gantry capacities are time-invariant, i.e., $C_{tg} = C_g$ for all $t = 1, \ldots, T$ and $g = 1, \ldots, G$. Then, there exists a stationary optimal solution to (B-LP(U)) in which the gantry capacity constraints are binding.*

Proof The dual (DB(U)) of problem (B-LP(U)) is

$$\text{minimize} \sum_{g=1}^{G} C_g \alpha_g + Uz$$

subject to (DB(U))

$$w_k \alpha_g + (\mu_k - \gamma_k) - \sum_{k' \in K} d_{k'} (\mu_{k'} - \gamma_{k'}) \geq r_k \qquad k \in K; g \in G \qquad (2.24)$$

$$z - \mu_k - \gamma_k \geq 0 \qquad k \in K \qquad (2.25)$$

$$\mu_k, \gamma_k \geq 0 \qquad k \in K$$

$$\alpha_g \geq 0 \qquad g \in G.$$

Due to dual constraint (2.24) and nonnegative α, we can write

$$\alpha_g \geq \frac{\sum_{k' \in K} d_{k'}(\mu_{k'} - \gamma_{k'}) - (\mu_k - \gamma_k) + r_k}{w_k} \qquad k \in K;\ g \in G$$

$$\alpha_g \geq \max_{k \in K} \left\{ \frac{\sum_{k' \in K} d_{k'}(\mu_{k'} - \gamma_{k'}) - (\mu_k - \gamma_k) + r_k}{w_k} \right\} \qquad g \in G$$

Since $0 \leq d_k \leq 1$ and $\sum_{k \in K} d_k = 1$, $\sum_{k' \in K} d_{k'}(\mu_{k'} - \gamma_{k'})$ is a weighted sum of $(\mu_k - \gamma_k)$ and we can write

$$\min_{k \in K} \{\mu_k - \gamma_k\} \leq \sum_{k' \in K} d_{k'}(\mu_{k'} - \gamma_{k'}) \leq \max_{k \in K} \{\mu_k - \gamma_k\}. \tag{2.26}$$

At least for one $k \in K$, $\sum_{k' \in K} d_{k'}(\mu_{k'} - \gamma_{k'}) - (\mu_k - \gamma_k) \geq 0$. In addition, r_k and w_k are positive for each $k \in K$. Hence, any feasible solution to problem (DB(U)) must have

$$\alpha_g > 0 \qquad g \in G$$

which implies that the optimal solution for (B-LP(U)) must have tight gantry capacity constraints

$$\sum_{k=1}^{K} w_k \bar{x}_{kg} = C_g \qquad g \in G.$$

□

By aggregating the $\bar{x}$-variables over all gantries it is easy to see that the capacity constraints in (BP-L(U)) reduces to one with a single gantry:

$$\sum_{k=1}^{K} w_k \sum_{g}^{G} \bar{x}_{kg} = \sum_{g}^{G} C_g = C. \tag{2.27}$$

Using the result of Lemma 2.5.1, the class of optimization problems to analyze (for a fixed total patient mix deviation U) when generating the Pareto frontier of (B-LP) is

$$z(U) = \text{maximize} \sum_{k=1}^{K} r_k X_k$$

subject to (F)

$$\sum_{k=1}^{K} w_k X_k = C \tag{2.28}$$

$$X_k - d_k \sum_{k'=1}^{K} X_{k'} = \bar{s}_k - \underline{s}_k \qquad k \in K \tag{2.29}$$

$$\sum_{k=1}^{K} (\underline{s}_k + \bar{s}_k) = U \tag{2.30}$$

$$X_k,\ \underline{s}_k,\ \bar{s}_k \geq 0 \qquad k \in K$$

where $z(U)$ denotes the optimal solution value to (F) for a fixed deviation U and $X_k = \sum_{g=1}^{G} \bar{x}_{kg}$ is the total number of patients treated in each category. We are particularly interested in U values between 0 and $\bar{U}$ where $\bar{U}$ is the largest value with the property that $z(U)$ is strictly incrusting. For convenience, the values of $\frac{r_k}{w_k}$ $(k \in K)$ are unique. Also, note that $\underline{s}_k$ $(\bar{s}_k)$ can be interpreted as the number of patients in category k in shortfall (excess) of the "ideal" number $d_k \sum_{k'=1}^{K} X_{k'}$. Interestingly, it can be shown that the total amount of patient mix shortfall/excess are equal to the same value. Lemma 2.5.2 shows how both the aggregate patient mix excess deviation and the aggregate patient mix shortfall deviation are both exactly half of the total deviation, $\frac{U}{2}$.

Lemma 2.5.2 *In problem (F), the cumulative shortfall and the cumulative excess patient mix deviation is equal to exactly half the allowable patient mix deviation. That is,*

$$\sum_{k \in K} \underline{s}_k = \sum_{k \in K} \bar{s}_k = \frac{U}{2}.$$

Proof From (2.30) we have that

$$\sum_{k \in K} \underline{s}_k = U - \sum_{k \in K} \bar{s}_k. \tag{2.31}$$

Furthermore, summing (2.29) over $k \in K$ yields

$$\sum_{k \in K} X_k - \sum_{k \in K} d_k \sum_{k'=1}^{K} X_{k'} = \sum_{k \in K} \bar{s}_k - \sum_{k \in K} \underline{s}_k. \tag{2.32}$$

Substituting (2.31) into (2.32) results in

$$\sum_{k\in K} \bar{s}_k - \left(U - \sum_{k\in K} \bar{s}_k \right) = \sum_{k\in K} X_k - \sum_{k\in K} d_k \sum_{k'=1}^{K} X_{k'} \tag{2.33}$$

$$= 0 \tag{2.34}$$

so

$$\sum_{k\in K} \bar{s}_k = \frac{U}{2}.$$

The result

$$\sum_{k\in K} \underline{s}_k = \frac{U}{2}$$

follows from the same argument.

□

Interestingly, this result indicates that (i) there is always some patient mix deviation in the optimal solution to (F) for $U > 0$ and (ii) the cumulative amount that the cumulative shortfall/excess is always the exact same known value. Using this result and numerous other structural properties of (F), the remainder of this section seeks to characterize the optimal solution to (F) for $U \in [0, \bar{U}]$.

2.5.1 Characterization of $z(U)$

The derivation of a Pareto optimal frontier involving the tradeoff between total fractions treated and allowable patient mix deviation equates to quantifying the change in the number of fractions treated as the allowable patient deviation mix is increased from $U = 0$ to the point in which any further increase in U does not lead to an increase in fractions treated. That is, we are interested in the structure of $z(U)$. Of course, in general, there are an infinite number of points on this frontier. Techniques, such as the NISE method, exist for generating this type of frontier, but those algorithms require the solution of multiple linear programs. However, since solving (F) for varying values of U equates to a perturbation of the right-hand-side (RHS) of a linear programming maximization problem, we know that $z(U)$ is piecewise linear nondecreasing and concave in U

(Bazaraa et al., 2011). An example of the structure of $z(U)$ is shown in Figure 2.1. We contend that the efficient frontier for our problem can be solved by utilizing the basis properties discussed in the next section to determine the finite number of breakpoints along our efficient front. As will be formally shown, our approach to analytically determining the frontier relies on the fact that (i) it is simple to determine an optimal basis at $z(\bar{U})$ and (ii) as U decreases, exactly one X or $\underline{s}$ variable is added/removed to/from the basis at a time. To begin our discussion of these results, the following section illustrates how applying established linear programming perturbation knowledge to (F) provides revealing properties of (i) the optimal basis at each breakpoint in our frontier and (ii) how this basis changes from breakpoint to breakpoint.

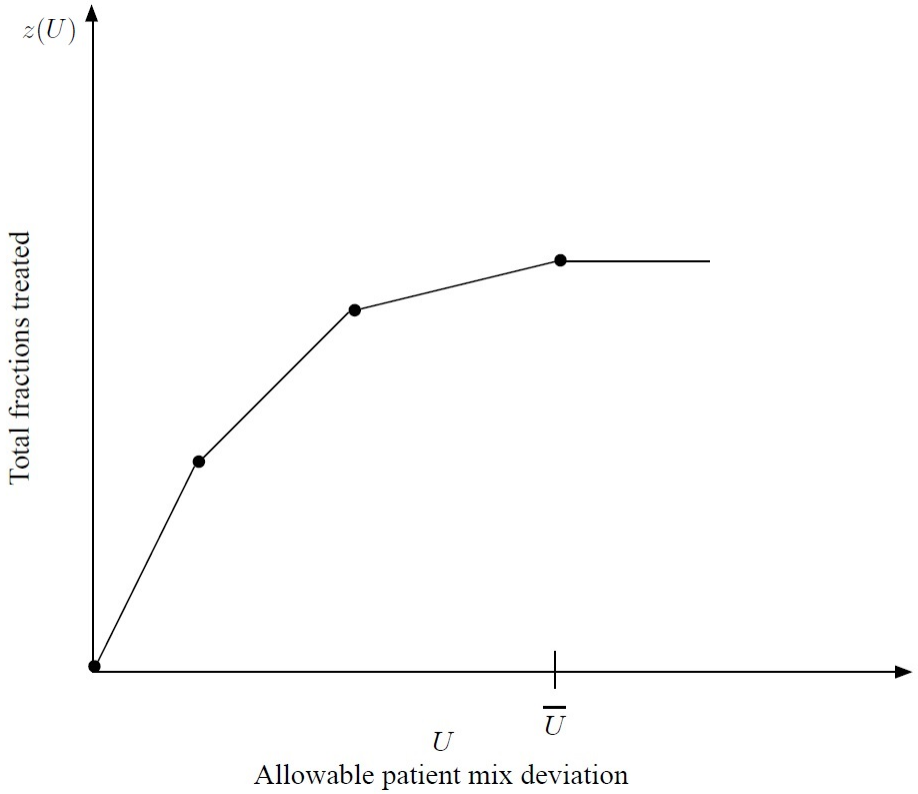

Figure 2.1: General structure: $z(U)$

2.5.2 Properties of an Optimal $z(U)$ Basis

As we will detail in the 2.5.3, an application of perturbation analysis to (F) aids in the revealing the efficient frontier (function $z(U)$) efficiently. Much of that analysis is driven by knowledge of

the optimal basis of (F) for any U in $0 < U < \bar{U}$. This section presents these optimal basis properties through the following two lemmas. Specifically, these lemmas reveal specific information regarding the so-called most attractive category, $\hat{k}$, where

$$\hat{k} = \arg\max_{k \in K} \left(\frac{r_k}{w_k} \right). \tag{2.35}$$

Recall that $\hat{k}$ is assumed to be unique. Lemma 2.5.3 shows that $X_{\hat{k}}$ is in the optimal basis of (F) with total deviation parameter $0 < U < \bar{U}$.

Lemma 2.5.3 *Let* β *be any basis associated with the optimal solution for problem F when* $0 < U < \bar{U}$. *Then,* $X_{\hat{k}} \in \beta$ *where* $\hat{k} = argmax_{k \in K}\{\frac{r_k}{w_k}\}$.

Proof Let β_0 be any basis associated with the optimal solution for problem F such that $X_{\hat{k}} \notin \beta_0$. Note that we can write $\underline{s}_{\hat{k}} = d_{\hat{k}} \sum_{k' \in \beta_0} X_{k'} - X_{\hat{k}} > 0$ since $\sum_{k' \in \beta_0} X_{k'} > 0$ and $X_{\hat{k}} \notin \beta_0$. Hence, $\underline{s}_{\hat{k}} \in \beta_0$. Now, we will check whether the dual feasibility is satisfied by β_0 which implies primal optimality. Thus, let DF be the dual of F as formulated below:

$$\text{minimize } C\alpha + Uq$$

subject to (DF)

$$w_k\alpha + \mu_k - \sum_{k' \in K} d_{k'}\mu_{k'} \geq r_k \qquad k \in K \qquad [X_k] \tag{2.36}$$

$$\mu_k + q \geq 0 \qquad k \in K \qquad [\underline{s}_k] \tag{2.37}$$

$$-\mu_k + q \geq 0 \qquad k \in K \qquad [\bar{s}_k] \tag{2.38}$$

From Lemma (2.5.2) there must exist at least one pair of basic variables $(X_{\underline{k}}, \bar{s}_{\underline{k}}) \in \beta_0$. Since, $X_{\hat{k}} \notin \beta_0$, we conclude that $\underline{k} \neq \hat{k}$. Moreover, we know that $X_{\underline{k}} > 0$ and $\bar{s}_{\underline{k}} > 0$. Then, from the

complementary slackness conditions, we can write

$$\mu_{\underline{k}} = q \tag{2.39}$$

$$\alpha = \frac{r_{\underline{k}} - \mu_{\underline{k}} + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{\underline{k}}} = \frac{r_{\underline{k}} - q + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{\underline{k}}}. \tag{2.40}$$

Similarly, for $\hat{k}$ such that $\underline{s}_{\hat{k}} > 0$ and $X_{\hat{k}} = 0$, we can also write the following dual constraints:

$$\mu_{\hat{k}} = -q \tag{2.41}$$

$$\alpha \geq \frac{r_{\hat{k}} - \mu_{\hat{k}} + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{\hat{k}}} = \frac{r_{\hat{k}} + q + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{\hat{k}}}. \tag{2.42}$$

We know that q is the dual variable associated with total deviation constraint (2.30) and can be interpreted as the slope of the efficient frontier for any U. Therefore, $q > 0$ when $0 < U < \bar{U}$ (see Figure 2.1). Recall that $\hat{k} = \arg\max_{k \in K}\{\frac{r_k}{w_k}\}$. Hence, α in (2.40) does not satisfy the inequality (2.42) which is necessary for the feasibility of (DF). By contradiction, we conclude that β_0 cannot be a basis associated with the optimal solution for problem (F) which also proves the lemma.

□

While the previous lemma specifies the existence of $X_{\hat{k}}$ variable in the optimal basis of (F), the following lemma focuses on the values of the shortfall and excess variables $(\underline{s}, \bar{s})$ in (F).

Lemma 2.5.4 *Let β be a basis associated with the optimal solution for problem (F) when $0 < U < \bar{U}$. Then, the following properties hold with regard to the shortfall and excess variables.*

(i) For each $k \in K$, at most one slack variables $(\bar{s}_k, \underline{s}_k)$ is strictly positive.

(ii) There is never deviation shortfall for patient category $\hat{k}$ (e.g. $\underline{s}_{\hat{k}} = 0$).

(iii) There is always deviation excess for patient category $\hat{k}$ (e.g. $\bar{s}_{\hat{k}} > 0$).

(iv) For all patient categories $k \neq \hat{k}$ there is no deviation excess (e.g. $\bar{s}_k = 0 \quad \forall k \in K, k \neq \hat{k}$).

Proof For (i), let $(\bar{s}_k, \underline{s}_k)$ be both positive, then the following complementary slackness conditions must be held.

$$\mu_k = -q \tag{2.43}$$

$$\mu_k = q \tag{2.44}$$

The only case conditions 2.43 and 2.44 are satisfied is when $q = 0$. Thus, if $q > 0$, at least one of the slack variable $(\bar{s}_k, \underline{s}_k)$ for each $k \in K$ has to be equal to zero.

For (ii), let β_0 be any basis associated with the optimal solution for problem (F) such that $\underline{s}_{\hat{k}} > 0$. We know that $X_{\hat{k}} \in \beta_0$ from Lemma 2.5.3. From Lemma 2.5.2, there must exist at least one patient category k'' with a pair of basic variables $(X_{k''}, \bar{s}_{k''}) \in \beta_0$ such that $X_{k''} > 0$ and $\bar{s}_{k''} > 0$. From (i) and $\underline{s}_{\hat{k}} > 0$, we conclude that $\bar{s}_{\hat{k}} = 0$ and therefore $k'' \neq \hat{k}$.

For k'', from the dual constraints (2.36) and (2.38), we can write

$$\alpha = \frac{r_{k''} - \mu_{k''} + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{k''}} = \frac{r_{k''} - q + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{k''}} \tag{2.45}$$

$$\mu_{k''} = q. \tag{2.46}$$

Similarly, for $\hat{k}$, we can write the dual constraint (2.37) as (2.47) and constraint (2.36) as (2.48) or (2.49).

$$\mu_{\hat{k}} = -q \tag{2.47}$$

$$\text{if } X_{\hat{k}} > 0, \quad \alpha = \frac{r_{\hat{k}} - \mu_{\hat{k}} + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{\hat{k}}} = \frac{r_{\hat{k}} + q + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{\hat{k}}} \tag{2.48}$$

$$\text{if } X_{\hat{k}} = 0, \quad \alpha \geq \frac{r_{\hat{k}} - \mu_{\hat{k}} + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{\hat{k}}} = \frac{r_{\hat{k}} + q + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{\hat{k}}} \tag{2.49}$$

From constraints (2.48) and (2.49), we see that $\alpha \geq \frac{r_{\hat{k}} + q + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{\hat{k}}}$. However, $\frac{r_{k''} - q + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{k''}} < \frac{r_{\hat{k}} + q + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{\hat{k}}}$ which violates the dual feasibility and hence, optimality of β_0. Therefore, by contradiction, we conclude that β_0 cannot be a basis associated with the optimal solution for problem F which also proves that $\underline{s}_{\hat{k}} = 0$.

For (iii), Let β_0 be any basis associated with the optimal solution for problem F such that $\bar{s}_{\hat{k}} = 0$. From Lemma 2.5.3 and (ii), we know that $X_{\hat{k}} \in \beta_0$ and $\underline{s}_{\hat{k}} = 0$, respectively. Thus, $X_{\hat{k}} = d_{\hat{k}} \sum_{k' \in \beta_0} X_{k'} > 0$ since $\underline{s}_{\hat{k}} = \bar{s}_{\hat{k}} = 0$ and $d_{\hat{k}} \sum_{k' \in K} X_{k'} > 0$. From Lemma 2.5.2, there must exist a patient category $k'' \neq \hat{k}$ with a pair of basic variables $(X_{k''}, \bar{s}_{k''}) \in \beta_0$ such that $X_{k''} > 0$ and $\bar{s}_{k''} > 0$. We conclude that $k'' \neq \hat{k}$ since $\bar{s}_{\hat{k}} = 0$. For k'', from the dual constraints (2.36) and (2.38), we can write

$$\mu_{k''} = q \tag{2.50}$$

$$\alpha = \frac{r_{k''} - \mu_{k''} + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{k''}} = \frac{r_{k''} - q + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{k''}}. \tag{2.51}$$

For $\hat{k}$, since $\underline{s}_{\hat{k}} = \bar{s}_{\hat{k}} = 0$, corresponding dual constraints (2.37) and (2.38) form a range on q as in (2.52). Moreover, since $X_{\hat{k}} > 0$, the dual constraint (2.36) is written as in (2.53).

$$-q \leq \mu_{\hat{k}} \leq q \tag{2.52}$$

$$\alpha = \frac{r_{\hat{k}} - \mu_{\hat{k}} + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{\hat{k}}} = \frac{r_{\hat{k}} - \mu_{\hat{k}} + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{\hat{k}}} \tag{2.53}$$

From (2.52) and (2.53), we can conclude that $\alpha \geq \frac{r_{\hat{k}} - q + \sum_{k' \in K} d_{k'} \mu_{k'}}{w_{k''}}$ which is violated by (2.51) since $\frac{r_{\hat{k}}}{w_{\hat{k}}} > \frac{r_{k''}}{w_{k''}}$. By contradiction, β_0 cannot be a basis associated with the optimal solution for problem F which also proves that $\bar{s}_{\hat{k}} > 0$.

For (iv), the argument follows similarly to (iii) by using the results of 2.5.3 and (iii).

□

With these results equipped to investigate (i) what the values of X_k for each point along our frontier and (ii) the manner in which X_k values change as we the total allowable deviation is decreased from $\bar{U}$ down to 0. The next section shows how the structural properties derived for (F) can be combined with established linear programming right-hand-side perturbation analysis to obtain the answers to each of these questions.

2.5.3 Revealing $z(U)$ via RHS Perturbation

Assume we had complete information about the optimal basis associated with some point on the efficient frontier. We address why this assumption is not restrictive in the following section. Berkelaar et al. (1997) show that this optimal basis, which we denote as β remains unchanged for some allowable change in the right-hand-side of the constraints in (F). They also conclude that this allowable range can be determined by solving the following optimization problem, F_β,

$$\text{maximize } \lambda$$

$$\text{subject to} \qquad (F_\beta)$$

$$\sum_{k:X_k\in\beta} w_k X_k = C \tag{2.54}$$

$$X_{\hat{k}} - d_{\hat{k}} \sum_{k':X_{k'}\in\beta} X_{k'} = \bar{s}_{\hat{k}} \tag{2.55}$$

$$X_k - d_k \sum_{k':X_{k'}\in\beta} X_{k'} = -\underline{s}_k \qquad k:(X_k,\underline{s}_k)\in\beta \tag{2.56}$$

$$X_k - d_k \sum_{k':X_{k'}\in\beta} X_{k'} = 0 \qquad k\neq\hat{k}:X_k\in\beta;\underline{s}_k\notin\beta \tag{2.57}$$

$$-d_k \sum_{k':X_{k'}\in\beta} X_{k'} = -\underline{s}_k \qquad k:X_k\notin\beta;\underline{s}_k\in\beta \tag{2.58}$$

$$\sum_{k:\underline{s}_k\in\beta} \underline{s}_k + \bar{s}_{\hat{k}} = U-\lambda \tag{2.59}$$

$$(\underline{s},\bar{s},\mathbf{X})\in\beta$$

where $\mathbf{X}$ is the collection of all $X_k\in\beta$. F_β is simply problem F with the the righti-hand-side U perturbed to be $U-\lambda$ and an objective to maximize λ using only the decision variables in the basis β. To accurately write this problem in terms of only decision variables in the basis β, (2.29) has been separated by categories that appear in the the basis β through (2.55)-(2.58). Notice that we take advantage of Lemma 2.5.4 to eliminate the need to consider $\underline{s}_k$ variables for category $\hat{k}$ and $\bar{s}_k$ variables for $k\neq\hat{k}$.

The following lemma shows that we can analytically obtain the optimal decision values to F_β. This will ultimately motivate the justification for how the basis changes along the efficient frontier as the total deviation decreases.

Lemma 2.5.5 *For any $0 < U < \bar{U}$ the optimal solution X decision values to the problem F_β are given as follows.*

$$X_k = d_k \sum_{k' \in K} X_{k'} \qquad X_k \in \beta,\ k \neq \hat{k} \tag{2.60}$$

$$X_k = 0 \qquad X_k \notin \beta,\ k \neq \hat{k} \tag{2.61}$$

$$X_{\hat{k}} = \left(d_{\hat{k}} + \sum_{k: X_k \notin \beta} d_k \right) \sum_{k' \in K} X_{k'}. \tag{2.62}$$

Furthermore,

$$\sum_{k' \in K} X_{k'} = \frac{C}{\left(d_{\hat{k}} + \sum_{k: X_k \notin \beta} d_k\right) w_{\hat{k}} + \sum_{k: X_k \in \beta, k \neq \hat{k}} w_k d_k}.$$

Proof The objective function can be rewritten by substituting (2.59) for λ to get

$$U - \text{minimize} \sum_{k: \underline{s}_k \in \beta} \underline{s}_k + \bar{s}_{\hat{k}}$$

subject to (F_β)

$$\sum_{k: X_k \in \beta} w_k X_k = C \tag{2.63}$$

$$X_{\hat{k}} - d_{\hat{k}} \sum_{k': X_{k'} \in \beta} X_{k'} = \bar{s}_{\hat{k}} \tag{2.64}$$

$$X_k - d_k \sum_{k': X_{k'} \in \beta} X_{k'} = -\underline{s}_k \qquad k : (X_k, \underline{s}_k) \in \beta \tag{2.65}$$

$$X_k - d_k \sum_{k': X_{k'} \in \beta} X_{k'} = 0 \qquad k \neq \hat{k} : X_k \in \beta; \underline{s}_k \notin \beta \tag{2.66}$$

$$-d_k \sum_{k': X_{k'} \in \beta} X_{k'} = -\underline{s}_k \qquad k : X_k \notin \beta; \underline{s}_k \in \beta \tag{2.67}$$

$$\sum_{k: \underline{s}_k \in \beta} \underline{s}_k + \bar{s}_{\hat{k}} = U - \lambda \tag{2.68}$$

$$(\underline{s}, \bar{s}, \mathbf{X}) \in \beta$$

which after substituting the values of s in the objective leaves.

$$U - \text{minimize} \left(X_{\hat{k}} - d_{\hat{k}} \sum_{k':X_{k'} \in \beta} X_{k'} \right) + \sum_{k:(X_k, \underline{s}_k) \in \beta} \left(d_k \sum_{k':X_{k'} \in \beta} X_{k'} - X_k \right) + \sum_{k:(X_k \notin \beta; \underline{s}_k \in \beta)} \left(d_k \sum_{k':X_{k'} \in \beta} X_{k'} \right)$$

subject to (F_β)

$$\sum_{k:X_k \in \beta} w_k X_k = C \tag{2.69}$$

$$X_{\hat{k}} \geq d_{\hat{k}} \sum_{k':X_{k'} \in \beta} X_{k'} \tag{2.70}$$

$$X_k \leq d_k \sum_{k:X_k \in \beta} X_{k'} \qquad k:(X_k, \underline{s}_k) \in \beta \tag{2.71}$$

$$X_k = d_k \sum_{k:X_k \in \beta} X_{k'} \qquad k \neq \hat{k}:X_k \in \beta; \underline{s}_k \notin \beta \tag{2.72}$$

$$X_k \geq 0 \qquad k: X_k \in \beta \tag{2.73}$$

$$(\underline{s}, \bar{s}, \mathbf{X}) \in \beta$$

However note that Lemma (2.5.2) states that

$$\sum_{k:\underline{s}_k \in \beta} \underline{s}_k + \bar{s}_{\hat{k}} = \frac{U - \lambda}{2} + \frac{U - \lambda}{2}. \tag{2.74}$$

Therefore the objective of F_β can be equivalently written as

$$U - \text{minimize } 2 \times \bar{s}_{\hat{k}} = U - 2 \left(X_{\hat{k}} - d_{\hat{k}} \sum_{k':X_{k'} \in \beta} X_{k'} \right). \tag{2.75}$$

Next, by substituting 2.55 into 2.54 we have

$$w_{\hat{k}} d_{\hat{k}} \sum_{k':X_{k'} \in \beta} X_{k'} + w_{\hat{k}} \bar{s}_{\hat{k}} + \sum_{k'' \neq \hat{k}:X_{k''} \in \beta} w_{k''} X_{k''} = C \tag{2.76}$$

or

$$\bar{s}_{\hat{k}} = \frac{C - w_{\hat{k}} d_{\hat{k}} \sum_{k':X_{k'} \in \beta} X_{k'} - \sum_{k' \neq \hat{k}:X_{k'} \in \beta} w_{k'} X_{k'}}{w_{\hat{k}}} \tag{2.77}$$

$$= \frac{C}{w_{\hat{k}}} - d_{\hat{k}} X_{\hat{k}} - \sum_{k' \neq \hat{k}:X_{k'} \in \beta} \left(\frac{w_{k'}}{w_{\hat{k}}} + d_{\hat{k}} \right) X_{k'} \tag{2.78}$$

Therefore, a final reformulation of F_β yields

$$U - \text{minimize } 2 \times \left(\frac{C}{w_{\hat{k}}} - d_{\hat{k}} X_{\hat{k}} - \sum_{k' \neq \hat{k}:X_{k'} \in \beta} \left(\frac{w_{k'}}{w_{\hat{k}}} + d_{\hat{k}} \right) X_{k'} \right)$$

subject to (F_β)

$$\sum_{k:X_k \in \beta} w_k X_k = C \tag{2.79}$$

$$X_{\hat{k}} \geq d_{\hat{k}} \sum_{k':X_{k'} \in \beta} X_{k'} \tag{2.80}$$

$$X_k \leq d_k \sum_{k:X_k \in \beta} X_{k'} \qquad k : (X_k, \underline{s}_k) \in \beta \tag{2.81}$$

$$X_k = d_k \sum_{k:X_k \in \beta} X_{k'} \qquad k \neq \hat{k} : X_k \in \beta; \underline{s}_k \notin \beta \tag{2.82}$$

$$X_k \geq 0 \qquad k : X_k \in \beta \tag{2.83}$$

$$(\underline{s}, \bar{s}, \mathbf{X}) \in \beta$$

Clearly, all objective coefficients for X_k ($k \neq \hat{k}$) are larger than that for $X_{\hat{k}}$. Therefore, each X_k ($k \neq \hat{k}$) in the basis will want to be satisfied at its desired mix level. That is,

$$X_k = d_k \sum_{k':X_{k'} \in \beta} X_{k'}. \tag{2.84}$$

However, we also want $X_{\hat{k}}$ to be as large as possible in order to reduce the objective further. To

get the value of $X_{\hat{k}}$ in F_β note that

$$\begin{aligned} X_{\hat{k}} &= \sum_{k':X_{k'}\in\beta} X_{k'} - \sum_{k:X_k\in\beta,k\neq\hat{k}} X_k = \sum_{k':X_{k'}\in\beta} X_{k'} - \sum_{k:X_k\in\beta,k\neq\hat{k}} d_k \sum_{k':X_{k'}\in\beta} X_{k'} \\ &= \sum_{k':X_{k'}\in\beta} X_{k'} \left(1 - \sum_{k:X_k\in\beta,k\neq\hat{k}} d_k\right) \\ &= \sum_{k'\in K} X_{k'} \left(d_{\hat{k}} + \sum_{k:X_k\notin\beta} d_k\right). \end{aligned}$$

Clearly, for all $k : X_k \notin \beta$, $X_k = 0$.

In order to calculate $\sum_{k'\in K} X_{k'}$, we plug the optimal X_k expressions just derived into the gantry capacity constraint. This leaves

$$\sum_{k\in K} w_k X_k = \sum_{k:X_k\in\beta,k\neq\hat{k}} w_k d_k \sum_{k'\in K} X_{k'} + w_{\hat{k}} \left(d_{\hat{k}} + \sum_{k:X_k\notin\beta} d_k\right) \sum_{k'\in K} X_{k'} = C$$

which yields

$$\sum_{k'\in K} X_{k'} = \frac{C}{\left(d_{\hat{k}} + \sum_{k:X_k\notin\beta} d_k\right) w_{\hat{k}} + \sum_{k:X_k\in\beta,k\neq\hat{k}} w_k d_k} \quad . \tag{2.85}$$

This completes the proof. □

In this section we have shown that the optimal solution to the problem F_β can be obtained analytically given that β is an optimal basis to the original problem F. In the following section, we show how both the origin and final points of the frontier can be derived and utilized as a starting point for the algorithm presented in Section 2.5.5.

2.5.4 Origin and End Point of $z(U)$

In this section, the extreme points, $z(0)$ and $z(\bar{U})$ of our efficient frontier. Lemma 2.5.6 states that the optimal solution value at these points can be analytically obtained.

Lemma 2.5.6 *The optimal solutions of (F) when $U = 0$ and $U = \bar{U}$ are given as follows.*

(i) When no deviation from the specified patient mix is allowed, the optimal solution value is

given by

$$z(0) = \frac{\sum_{k \in K} r_k d_k}{\sum_{k \in K} w_k d_k} C$$

and then number of patients treated in each category is

$$X_k = \frac{d_k}{\sum_{k' \in K} w_{k'} d_{k'}} C \ (k \in K)$$

.

(ii) When $U = \bar{U}$,

$$z(\bar{U}) = \frac{r_{\hat{k}} C}{w_{\hat{k}}}$$

and the number of patients treated in each category is

$$X_k = 0 \qquad k \in K, k \neq \hat{k} \tag{2.86}$$

$$X_{\hat{k}} = \frac{C}{w_{\hat{k}}} \tag{2.87}$$

where $\hat{k} = \arg\max_{k \in K} \{\frac{r_k}{w_k}\}$

Proof The arguments for the two results are as follows.

(i) See section 2.4.3.

(ii) By definition, when $U = \bar{U}$ the amount of deviation is large enough such that no additional benefit can be obtained by allowing additional deviation. Therefore, only the gantry capacity constraints (2.28) are active. Notice that (F) with only 2.28 reduces to the well-known linear relaxation of a standard knapsack problem (LP-KP). An optimal solution to (LP-KP) is known to exist for which only a single item (patient category) is included at a non-zero level. Furthermore, this item is known to have the largest ratio of profit to item size (e.g. r_k to w_k).

Given this result, we know the optimal basis for the last point along the efficient frontier before no further improvement can be reached by considering addition deviation. We use this result, along with the results regarding the basis changes along the frontier in the algorithm proposed in the following section.

2.5.5 Efficient Frontier Generation Algorithm

We summarize the findings of the previous sections as below:

- According to Berkelaar et al. (1997), the optimal solution of F_β addresses the transition point of a linearity interval.

- Note that the optimal solution to problem F when $U \geq \bar{U}$ (or $q = 0$) is given in 2.86 and 2.87. Thus, one of the optimal basis would be $B_0 = \{X_{\hat{k}} \cup \bar{s}_{\hat{k}} \cup \underline{s}_k : k \in K\}$. Once U is decreased to $\bar{U}$ (transition point), $\underline{s}_{\hat{k}}$ drops to zero and therefore has to be replaced with a nonbasic variable. This is called parametric perturbation analysis with optimal basis in Bazaraa et al. (2011).

  - At this point, $\bar{s}_k \ \forall k \in K, k \neq \hat{k}$ cannot be in β when $0 < U < \bar{U}$ (Lemma 2.5.4) and $\underline{s}_k = 0 \ \forall k \in K, k \neq \hat{k}$ are already in β. Therefore, $X_k \ \forall k \in K, k \neq \hat{k}$ are the only possible entering variables for the second to last linearity interval. However, $\underline{s}_k = 0 \ ; \forall k \in K, k \neq \hat{k}$ variables can be potential entering variables at the future breakpoints.

  - We utilize the optimal solution of F_β when $U < \bar{U}$ in order to avoid solving another LP to calculate the total deviation and fraction levels at the next breakpoint. We calculate $q = \frac{\text{change in total fraction between breakpoints}}{\text{change in total deviation between breakpoints}}$ for each nonbasic (entering candidate) X_k and $\underline{s}_k \ \forall k \in K, k \neq \hat{k}$. Then, we select the nonbasic variable with minimum q as an entering variable if the resulting $\lambda \geq 0$. Based on the optimal solution structure of the breakpoints from Lemma 2.5.5,

    (i) if $X_{k''} \ (k'' \in K, k'' \neq \hat{k})$ is the entering variable, $\underline{s}_{k''} \ (k'' \in K, k'' \neq \hat{k})$ must leave the basis at the next breakpoint or

    (ii) if $\underline{s}_{k''} \ (k'' \in K, k'' \neq \hat{k})$ is the entering variable, $X_{k''} \ (k'' \in K, k'' \neq \hat{k})$ must leave the basis at the next breakpoint.

  - We generalize this procedure by relying on the fact that $(X_k, \underline{s}_k)$ variables such that $k \in K, k \neq \hat{k}$ are the only candidates to (leave) enter into the optimal basis at the transition

points.

The above discussion motivates an algorithm to generate the entire efficient frontier analytically. In order to calculate all breakpoints of the efficient frontier, we propose Algorithm 1. Below is the extra notation used in Algorithm 1.

- $j \in J$, set of efficient breakpoints
- $k \in N_X$, set of patient categories with nonbasic X_k variables
- $k \in N_{\underline{S}}$, set of patient categories with nonbasic $\underline{s}_k$ variables except $\hat{k}$ ($\hat{k} \notin N_{\underline{S}}$)
- $k \in P_\beta$, set of patient categories with basic X_k variables except $\hat{k}$ ($\hat{k} \notin P_\beta$ and $K = \{\hat{k} \cup N_X \cup P_\beta\}$)
- X_k^j, the value of X_k at Pareto breakpoint j
- $TD^j = \sum_{k \in K} \left| d_k \sum_{k' \in K} X_{k'}^j - X_k^j \right|$, total deviation at Pareto breakpoint j
- $TF^j = \sum_{k \in K} r_k X_k^j$, total fraction at Pareto breakpoint j
- $XY_k^{k''}$, the value of X_k at the next breakpoint if $X_{k''}$ is selected to enter into the basis
- $SY_k^{k''}$, the value of X_k at the next breakpoint if $\underline{s}_{k''}$ is selected to enter into the basis
- $XtempTD^{k''}$, total deviation at the next breakpoint if $X_{k''}$ is selected to enter into the basis
- $XtempTF^{k''}$, total fraction at the next breakpoint if $X_{k''}$ is selected to enter into the basis
- $StempTD^{k''}$, total deviation at the next breakpoint if $\underline{s}_{k''}$ is selected to enter into the basis
- $StempTF^{k''}$, total fraction at the next breakpoint if $\underline{s}_{k''}$ is selected to enter into the basis
- $q^{X_{k''}} = \frac{\text{change in total fraction}}{\text{change in total deviation}}$, the value of the dual variable q (slope) if $X_{k''}$ is selected to enter into the basis
- $q^{\underline{S}_{k''}} = \frac{\text{change in total fraction}}{\text{change in total deviation}}$, the value of the dual variable q (slope) if $\underline{s}_{k''}$ is selected to enter into the basis

Algorithm 1 Generates efficient frontier breakpoints

Initialize:

Set $j = 1$, $\hat{k} = argmax_{k \in K} \left\{ \frac{r_k}{w_k} \right\}$, $M =$ Sufficiently large number

Set $N_X = K \setminus \{\hat{k}\}$, $N_{\underline{S}} = \emptyset$, $P_\beta = \emptyset$, $X_{\hat{k}}^j = \frac{C}{w_{\hat{k}}}$, $X_k^j = 0$ for each $k \in N_X$

Calculate $TD^j = \sum_{k \in K} \left| d_k \sum_{k' \in K} X_{k'}^j - X_k^j \right|$ and $TF^j = \sum_{k \in K} r_k X_k^j$

while $N_X \neq \emptyset$ **do**

  j=j+1

  for all $k \in N_X$ **do**

    Select $k'' = k$

    $Y_{\hat{k}}^{X^{k''}} = C \left(\frac{d_{\hat{k}} + \sum_{k \in N_X \setminus \{k''\}} d_k}{\left(d_{\hat{k}} + \sum_{k \in N_X \setminus \{k''\}} d_k \right) w_{\hat{k}} + \sum_{k \in P_\beta \cup \{k''\}} d_k w_k} \right)$

    $Y_{k''}^{X^{k''}} = C \left(\frac{d_{k''}}{\left(d_{\hat{k}} + \sum_{k \in N_X \setminus \{k''\}} d_k \right) w_{\hat{k}} + \sum_{k \in P_\beta \cup \{k''\}} d_k w_k} \right)$

    $Y_k^{X^{k''}} = C \left(\frac{d_k}{\left(d_{\hat{k}} + \sum_{k \in N_X \setminus \{k''\}} d_k \right) w_{\hat{k}} + \sum_{k \in P_\beta \cup \{k''\}} d_k w_k} \right)$ for all $k \in P_\beta$

    $Y_k^{X^{k''}} = 0$ for all $k \in N_X \setminus \{k''\}$

    $\sum_{k \in K} Y_k^{X^{k''}} = C \left(\frac{1}{\left(d_{\hat{k}} + \sum_{k \in N_X \setminus \{k''\}} d_k \right) w_{\hat{k}} + \sum_{k \in P_\beta \cup \{k''\}} d_k w_k} \right)$

    $XtempTD^{k''} = \sum_{k \in K} \left| d_k \sum_{k' \in K} Y_{k'}^{X^{k''}} - Y_k^{X^{k''}} \right|$, $XtempTF^{k''} = \sum_{k \in K} r_k Y_k^{X^{k''}}$

    if $(TD^{j-1} - XtempTD^{k''} > 0)$ **then**

      $q^{X_{k''}} = \frac{TF^{j-1} - XtempTF^{k''}}{TD^{j-1} - XtempTD^{k''}}$

    else

      $q^{X_{k''}} = M$

    end if

  end for

  for all $k \in N_{\underline{S}}$ **do**

    Select $k'' = k$

    $Y_{\hat{k}}^{\underline{S}_{k''}} = C \left(\frac{d_{\hat{k}} + \sum_{k \in N_X \cup \{k''\}} d_k}{\left(d_{\hat{k}} + \sum_{k \in N_X \cup \{k''\}} d_k \right) w_{\hat{k}} + \sum_{k \in P_\beta \setminus \{k''\}} d_k w_k} \right)$

    $Y_k^{\underline{S}_{k''}} = C \left(\frac{d_k}{\left(d_{\hat{k}} + \sum_{k \in N_X \cup \{k''\}} d_k \right) w_{\hat{k}} + \sum_{k \in P_\beta \setminus \{k''\}} d_k w_k} \right)$ for all $k \in P_\beta$

    $Y_k^{\underline{S}_{k''}} = 0$ for all $k \in N_X \cup \{k''\}$

    $\sum_{k \in K} Y_k^{\underline{S}_{k''}} = C \left(\frac{1}{\left(d_{\hat{k}} + \sum_{k \in N_X \cup \{k''\}} d_k \right) w_{\hat{k}} + \sum_{k \in P_\beta \setminus \{k''\}} d_k w_k} \right)$

    $StempTD^{k''} = \sum_{k \in K} \left| d_k \sum_{k' \in K} Y_{k'}^{\underline{S}_{k''}} - Y_k^{\underline{S}_{k''}} \right|$, $StempTF^{k''} = \sum_{k \in K} r_k Y_k^{\underline{S}_{k''}}$

    if $(TD^{j-1} - StempTD^{k''} > 0)$ **then**

      $q^{\underline{S}_{k''}} = \frac{TF^{j-1} - StempTF^{k''}}{TD^{j-1} - StempTD^{k''}}$

    else

      $q^{\underline{S}_{k''}} = M$

    end if

  end for

if $(\min_{k\in N_X}\{q^{X_k}\} \leq \min_{k\in N_{\underline{S}}}\{q^{\underline{S}_k}\})$ **then**
  $k^* = argmin_{k\in N_X}\{q^{X_k}\}$
  Set $X_k^j = Y_k^{X_{k^*}}$ for each $k \in K$
  Set $TD^j = XtempTD^{k^*}$ and $TF^j = XtempTF^{k^*}$
  Set $N_X = N_X \setminus \{k^*\}$, $P_\beta = P_\beta \cup \{k^*\}$ and $N_{\underline{S}} = N_{\underline{S}} \cup \{k^*\}$
else
  $k^* = argmin_{k\in N_{\underline{S}}}\{q^{\underline{S}_k}\}$
  Set $X_k^j = Y_k^{\underline{S}_{k^*}}$ for each $k \in K$
  Set $TD^j = StempTD^{k^*}$ and $TF^j = StempTF^{k^*}$
  Set $N_X = N_X \cup \{k^*\}$, $P_\beta = P_\beta \setminus \{k^*\}$ and $N_{\underline{S}} = N_{\underline{S}} \setminus \{k^*\}$
end if
end while

Starting form the last breakpoint, Algorithm 1 enumerates all possible breakpoints that are not necessarily to be on the Pareto frontier as long as there exists a decrease on U (i.e. positive λ at the solution of F_β). Entering variable with the minimum slope (q) points to the linearity interval and the solution of F_β with the updated basis identifies the next breakpoint at the end of this frontier segment. The algorithm iterates until all X variables become basic ($N_X = \emptyset$). Solution of F_β with all basic X variables is identical to the solution of $z(0)$ given in Lemma 2.5.6 where $U = 0$.

2.5.6 Computational Study

This section assesses the efficiency gains due to our algorithm that is built on several mathematical structures in achieving the entire efficient frontier of the bicriteria base patient scheduling problem (*BB*). The original formulation of (*BB*) in Gedik (2011) and Section 2.3.2 is solved by the *NISE* method Cohon et al. (1979). We refer the reader to Gedik (2011) for a very detailed explanation of the NISE method application on (*BB*). It requires solving several LPs with weighted objective function to generate efficient frontier line segments. Therefore, these LPs are modeled in IBM ILOG CPLEX Optimization Studio 12.6 which uses IBM ILOG CPLEX 12.6 solver. IBM ILOG Concert Technology is utilized for embedding the formulations in C++ language into IBM ILOG CPLEX. Algorithm 1 is also implemented in C++ programming language. We run all

test problems on a Core 2 Duo 2.93 GHz, 16 GB RAM computer.

Same set of problem parameters used in Gedik (2011) is employed to create problem instances. Table 2.2 lists three different patient mix ratios (PMRs) and other input parameters for each patient category. In addition, we use two different capacity levels of gantries ($C_{tg} = C_g =$ 720 and 900 minutes) and ten different planning period lengths ($|T| = 100, 200, 300, \ldots, 1000$ days) to assess the effect of varying daily gantry capacity and planning period length.

Table 2.2: Parameters

	Patient Categories									
	1	2	3	4	5	6	7	8	9	10
PMR1	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10
PMR2	0.20	0.10	0.20	0.10	0.05	0.10	0.05	0.10	0.05	0.05
PMR3	0.65	0.15	0.07	0.03	0.03	0.02	0.01	0.01	0.02	0.01
f_k (# of fractions per day)	1	1	2	2	1	1	1	1	1	1
n_k (min)	40	40	31	31	30	30	30	30	42	12
c_k (min)	18	30	35	45	35	55	60	90	50	35
$\bar{c}_k$ (min)	15	15	20	25	20	20	45	45	30	20

Table 2.3 and 2.4 demonstrate computation times to generate the efficient frontier of problem (*BB*) by NISE method and our algorithm with the problem instances provided in 2.2. The only difference between the problem instances in Table 2.3 and 2.4 is that c_k and $\bar{c}_k$ are randomly generated for each patient category in Table 2.4 by a discrete uniform distribution with range $[18, 90]$ and $[15, 45]$, respectively. We note that efficient frontiers for all instances generated by the NISE method and our algorithm have identical breakpoints and linear segments. On the other hand, the average computation time for an efficient frontier required by NISE method exceeds that of the algorithm by a factor of 1700 in Table 2.3 and 1800 in Table 2.4. This is a direct result of providing a time invariant equivalent problem formulation for (*BB*) and later the characteristics of the optimal solution to this version. Note that the number of decision variables in (*BB*) increases as $|T|$ increases. Thus, the efficient frontier generation time by the NISE method for (*BB*) tends to increase gradually as ($|T|$) gets larger. On the other hand, computation times for our algorithm do not change by increasing $|T|$ since it is able to identify time invariant efficient

frontiers for (*BB*). Another advantage of our approach over the NISE method is that we generate *exact* efficient frontiers of the bicriteria base problem, whereas there is a marginal error embedded in the NISE method to approximate the exact efficient frontiers.

Table 2.3: Computation Time of Algorithm 1 and NISE Method

C_{tg} (min.)	$\|T\|$ (days)	Patient Mix Ratio	NISE Method Time (sec,)	Algorithm Time (sec.)	C_{tg} (min.)	$\|T\|$ (days)	Patient Mix Ratio	NISE Method Time (sec,)	Algorithm Time (sec.)
720	100	PMR 1	42.99	0.50	720	600	PMR 1	897.03	0.50
720	100	PMR 2	44.02	0.50	720	600	PMR 2	790.06	0.17
720	100	PMR 3	51.34	0.18	720	600	PMR 3	883.86	0.48
900	100	PMR 1	41.99	0.41	900	600	PMR 1	931.93	0.11
900	100	PMR 2	45.32	0.49	900	600	PMR 2	1122.86	0.50
900	100	PMR 3	51.97	0.50	900	600	PMR 3	822.21	0.61
720	200	PMR 1	104.20	0.46	720	700	PMR 1	1126.89	0.55
720	200	PMR 2	104.68	0.11	720	700	PMR 2	905.45	0.52
720	200	PMR 3	105.77	0.50	720	700	PMR 3	1021.57	0.55
900	200	PMR 1	97.01	0.51	900	700	PMR 1	1104.55	0.49
900	200	PMR 2	100.98	0.46	900	700	PMR 2	915.62	0.50
900	200	PMR 3	106.59	0.50	900	700	PMR 3	1039.34	0.55
720	300	PMR 1	239.52	0.50	720	800	PMR 1	1489.06	0.44
720	300	PMR 2	235.86	0.49	720	800	PMR 2	1150.34	0.27
720	300	PMR 3	236.15	0.13	720	800	PMR 3	1249.56	0.54
900	300	PMR 1	237.07	0.50	900	800	PMR 1	1448.11	0.45
900	300	PMR 2	248.09	0.50	900	800	PMR 2	1195.86	0.53
900	300	PMR 3	229.80	0.51	900	800	PMR 3	1293.99	0.54
720	400	PMR 1	397.29	0.50	720	900	PMR 1	1804.78	0.53
720	400	PMR 2	343.59	0.44	720	900	PMR 2	1546.92	0.55
720	400	PMR 3	394.84	0.51	720	900	PMR 3	1653.72	0.55
900	400	PMR 1	393.45	0.50	900	900	PMR 1	1886.67	0.54
900	400	PMR 2	353.88	0.42	900	900	PMR 2	1582.47	0.54
900	400	PMR 3	400.00	0.51	900	900	PMR 3	1759.52	0.55
720	500	PMR 1	584.03	0.51	720	1000	PMR 1	2305.90	0.55
720	500	PMR 2	513.34	0.50	720	1000	PMR 2	2213.82	0.58
720	500	PMR 3	608.02	0.50	720	1000	PMR 3	1979.58	0.55
900	500	PMR 1	572.06	0.50	900	1000	PMR 1	2335.11	0.40
900	500	PMR 2	516.41	0.40	900	1000	PMR 2	2098.16	0.54
900	500	PMR 3	593.15	0.51	900	1000	PMR 3	2138.74	0.53

Table 2.4: Computation Time of Algorithm 1 and NISE Method with randomly generated c_k and $\bar{c}_k$

C_{tg} (min.)	$\|T\|$ (days)	Patient Mix Ratio	NISE Method Time (sec,)	Algorithm Time (sec.)	C_{tg} (min.)	$\|T\|$ (days)	Patient Mix Ratio	NISE Method Time (sec,)	Algorithm Time (sec.)
720	100	PMR 1	36.81	0.53	720	600	PMR 1	2034.03	0.52
720	100	PMR 2	39.55	0.53	720	600	PMR 2	787.58	0.49
720	100	PMR 3	42.66	0.53	720	600	PMR 3	849.61	0.53
900	100	PMR 1	36.10	0.52	900	600	PMR 1	1886.61	0.52
900	100	PMR 2	39.94	0.50	900	600	PMR 2	729.43	0.53
900	100	PMR 3	42.17	0.52	900	600	PMR 3	819.16	0.51
720	200	PMR 1	108.09	0.52	720	700	PMR 1	950.76	0.54
720	200	PMR 2	107.47	0.52	720	700	PMR 2	971.66	0.53
720	200	PMR 3	121.42	0.45	720	700	PMR 3	1056.52	0.50
900	200	PMR 1	105.11	0.53	900	700	PMR 1	967.36	0.52
900	200	PMR 2	106.87	0.49	900	700	PMR 2	971.07	0.17
900	200	PMR 3	201.01	0.52	900	700	PMR 3	1122.71	0.52
720	300	PMR 1	218.59	0.52	720	800	PMR 1	1292.23	0.52
720	300	PMR 2	228.36	0.33	720	800	PMR 2	1280.77	0.52
720	300	PMR 3	249.65	0.52	720	800	PMR 3	1362.53	0.51
900	300	PMR 1	222.10	0.49	900	800	PMR 1	1333.92	0.52
900	300	PMR 2	231.47	0.50	900	800	PMR 2	1333.62	0.52
900	300	PMR 3	250.03	0.52	900	800	PMR 3	1340.50	0.52
720	400	PMR 1	361.71	0.18	720	900	PMR 1	1655.96	0.53
720	400	PMR 2	384.41	0.43	720	900	PMR 2	1745.09	0.52
720	400	PMR 3	425.67	0.53	720	900	PMR 3	1775.85	0.38
900	400	PMR 1	363.66	0.52	900	900	PMR 1	1943.32	0.52
900	400	PMR 2	387.72	0.52	900	900	PMR 2	1769.37	0.53
900	400	PMR 3	409.59	0.52	900	900	PMR 3	1764.01	0.48
720	500	PMR 1	577.98	0.45	720	1000	PMR 1	2269.01	0.49
720	500	PMR 2	626.06	0.52	720	1000	PMR 2	2139.59	0.53
720	500	PMR 3	700.93	0.45	720	1000	PMR 3	2222.67	0.50
900	500	PMR 1	536.41	0.53	900	1000	PMR 1	2123.17	0.47
900	500	PMR 2	602.34	0.52	900	1000	PMR 2	2156.84	0.47
900	500	PMR 3	639.11	0.53	900	1000	PMR 3	2272.29	0.48

2.6 Conclusions and Future Research

In this chapter, we study the characteristics of optimal solution for the strategic level patient scheduling problem with patient mix restrictions. After showing the existence of time invariant feasible solution with identical objective function value to any time variant (bicriteria) base model feasible solution with stationary gantry capacities, we focus on exploring the optimal solutions for the base model with and without deviation from the desired patient mix levels. In case of no deviation, our efforts illustrate how to obtain the analytically derived optimal solution for the base model with several side constraints. These findings provide very useful insights for the healthcare decision makers in practice. Below, we provide several observations we learned by studying the optimal solutions of (B) with strict patient mix restrictions.

- Maximizing the total reward (number of treatment sessions) is equivalent to maximizing the number of patients treated.

- The optimal solution to the base model with or without gantry switching flexibility can be interpreted as

$$\text{Total gantry capacity} \times \frac{\text{Weighted total treatment sessions (reward)}}{\text{Weighted total treatment time}}.$$

- The optimal solution to the base model with anesthesia patients restrictions can be interpreted as

$$\text{Weighted total treatment sessions (reward)} \times \min\{GC, AC\}$$

where

$$GC = \frac{\text{Total gantry capacity}}{\text{Weighted total treatment time}}$$

$$AC = \frac{\text{Total anesthesia availability in all gantries}}{\text{Weighted total treatment time for anesthesia patients}}$$

.

- The optimal solution to the base model with BID patients restrictions can be interpreted as

$$\text{Weighted total treatment sessions (reward)} \times \min\{GC, BC\}$$

where

$$BC = \frac{\text{Total BID patients availability in all gantries}}{\text{Weighted total treatment time for BID patients}}$$

.

By utilizing these simple rule of thumbs, scenario analysis can be performed very easily regarding desired percent mix levels, anesthesia/BID team availability, treatment time and reward for treating patient types. Clearly, these analytical results save the decision makers from solving an optimization problem to evaluate the impacts of a patient mix percentages as well as other capacity adjustments.

Another key contributions of this chapter is the derivation of the optimal solution for the bicriteria base patient scheduling model (*BB*). As shown in Table 2.3 and Table 2.4, our algorithm provides a very fast way of assessing the impacts of patient mix selection on the two objectives; total fractions delivered to patients and corresponding deviation from patient mix preferences in a planning period. Managers and facility administration utilize patient mix preferences to estimate approximate resource consumption, costs and revenues based on the values defined by DRGs. A strategic level capacity plan can be easily generated by producing exact efficient frontiers by using our approach that relate the treated total number of patients to the total deviation from patient mix requirements.

Of course, since LP efficient frontiers tend to schedule a fractional amount of patients, they can only be utilized for a strategic plan to observe the fundamental behaviors of the problem components under different operational restrictions. It remains desirable to determine a frontier of integer solutions that reflect the exact number of scheduled patients required from each category k in each gantry g on each day t. The efficient frontier of (*BB*) can be used to approximate the exact IP frontier. In order to investigate the quality of this approximation, initially, we can

constraint on one objective (e.g. total fractions) associated with the breakpoints of the LP efficient frontier and solve an integer program to optimize the remaining decision (e.g. patient mix deviation).

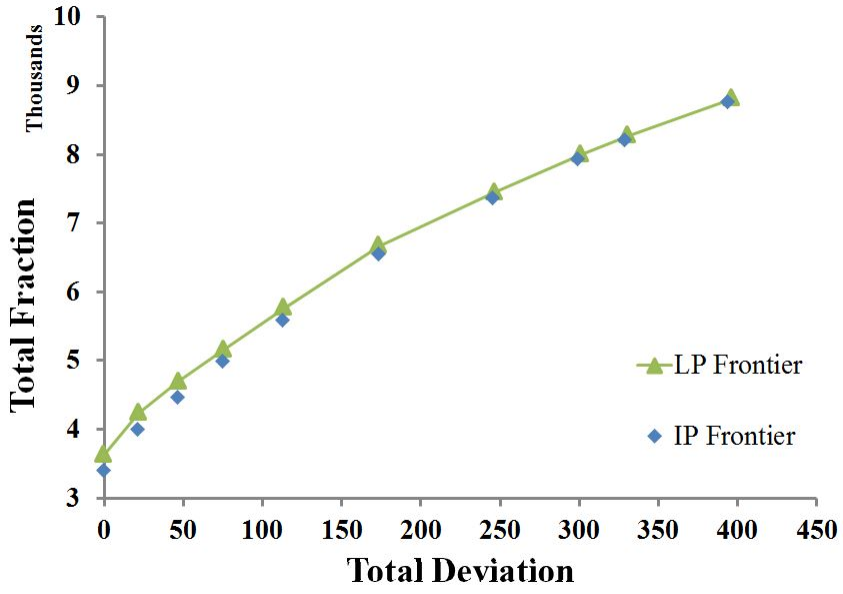

Figure 2.2: Bicriteria base model IP and LP frontiers with PMR 1

Figure 2.2 is one example of the IP efficient frontier points obtained by using this method. For small total deviation amounts (less than 50), the IP efficient points reveal that far fewer total fractions can be treated than in the case of the LP frontier. This suggests that as the patient mix deviation restriction is more strictly enforced, the approximation of the IP frontier by the LP is less desirable. Obviously, solving an IP for each breakpoint is computationally expensive and the breakpoints of the LP frontier do not necessarily coincide with that of the IP frontier. However, LP frontier still provides a bound on IP frontier and can be more efficiently constructed by our algorithm. As an immediate future research direction towards efficiently constructing IP frontier, we would like to utilize some information revealed through analyzing the characteristics of the efficient frontier for (*BB*). For instance, the total number of patients across all gantries associated with a breakpoint is calculated analytically. This can be embedded in the IP formulation of (*BB*) as a bound on the integer number of patients that can be accepted to the facility.

The second research direction is to expand the findings of Section 2.5 to explore efficient frontiers for the (*BB*) problem with the side constraints. Based on the type of the side constraint added to the (*BB*), further investigations must be tailored to identify the impacts of additional restriction(s) on the efficient frontier. For instance, if anesthesia or BID side constraints (inequalities) are imposed into (*BB*), it is extremely vital to be able to monitor which constraint (side or gantry capacity) remains binding as U takes on different values. Identifying this relationship is even important when U is sufficiently large since $\hat{k} = argmax_{k \in K}\{\frac{r_k}{w_k}\}$ might belong to a special category (BID or anesthesia) and therefore limit the total number of patients. In this case, we need to be able to monitor which inequality will become active and when. If we assume no degenerate solution on efficient frontier segments except breakpoints, then any change in the status of an inequality (becoming binding or unbinding) will create extra breakpoint since a slack variable becomes either nonbasic or basic.

Finally, it still remains desirable to demonstrate the impacts of not having unique $\frac{r_k}{w_k}$ ratios across all patient categories on efficient frontiers.

Appendix

2.A Work Verification Letter

College of Engineering
Department of Industrial Engineering

Date: July 8, 2014

Graduate School
University of Arkansas

Dear Dr. Needy:

I am writing to verify that Ridvan Gedik completed more than 51% of the work for the chapter titled "Strategic Level Patient Scheduling Mathematical Modeling in a Proton Therapy Facility," in his dissertation. He is also the first author of this article.

Sincerely,

Chase Rainwater
cer@uark.edu
479-575-2687
Assistant Professor
Department of Industrial Engineering
University of Arkansas

Bibliography

Adan, I., Bekkers, J., Dellaert, N., Vissers, J., and Yu, X. (2009). Patient mix optimisation and stochastic resource requirements: A case study in cardiothoracic surgery planning. *Health Care Management Science*, 12(2):129–141.

Adan, I. and Vissers, J. (2002). Patient mix optimisation in hospital admission planning: a case study. *International Journal of Operations & Production Management*, 22(4):445–461.

Averill, R. F., Muldoon, J. H., Vertrees, J. C., Goldfield, N. I., Mullin, R. L., Fineran, E. C., Zhang, M. Z., Steinbeck, B., and Grant, T. (1998). The evolution of casemix measurement using diagnosis related groups (drgs). *Wallingford: 3M Health Information Systems*.

Ballard, S. M. and Kuhl, M. E. (2006). The use of simulation to determine maximum capacity in the surgical suite operating room. *Proceedings of the 2006 Winter Simulation Conference*, pages 433–438.

Bazaraa, M. S., Jarvis, J. J., and Sherali, H. D. (2011). *Linear Programming and Network Flows*. Wiley.

Berkelaar, A. B., Roos, K., and Terlaky, T. (1997). *The optimal set and optimal partition approach to linear and quadratic programming*. Springer.

Blake, J. T. and Carter, M. W. (2002). A goal programming approach to strategic resource allocation in acute care hospitals. *European Journal of Operational Research*, 140(3):541 – 561.

Bosch, P. M. V. and Dietz, D. C. (2000). Minimizing expected waiting in a medical appointment system. *IIE Transactions*, 32:841–848.

Bowers, J. and Mould, G. (2005). Ambulatory care and orthopaedic capacity planning. *Health Care Management Science*, 8:41–47.

Cardoen, B. and Demeulemeester, E. (2008). Capacity of clinical pathways a strategic multi-level evaluation tool. *Journal of Medical Systems*, 32:443–452.

Cardoen, B., Demeulemeester, E., and Bëlin, J. (2010). Operating room planning and scheduling: A literature review. *European Journal of Operation Research*, 201:921–932.

Cayirli, T., Veral, E., and Rosen, H. (2006). Designing appointment scheduling systems for ambulatory care services. *Health Care Management Science*, 9:47–58.

Chaabane, S., Meskens, N., Guinet, A., and Laurent, M. (2008). Comparison of two methods of operating theatre planning: Application in belgian hospitals. *Journal of Systems Science and Systems Engineering*, 17(2):171–186.

Cohon, J. L., Church, R. L., and Sheer, D. N. (1979). Generating multiobjective trade-offs: An algorithm for bicriterion problems. *Water Resources Research*, 19:1001–1010.

Conforti, D., Guerriero, F., and Guido, R. (2008). Optimization models for radiotherapy patient scheduling. *4OR*, 6:263–278.

Conforti, D., Guerriero, F., and Guido, R. (2010). Non-block scheduling with priority for radiotherapy treatments. *European Journal of Operation Research*, 201:289–296.

Gedik, R. (2011). *Evaluating the Capacity of a Proton Therapy Facility*. University of Arkansas, Fayetteville.

Hughes, W. L. and Soliman, S. Y. (1985). Short-term case mix management with linear programming. *Hospital and Health Services Administration*, 30(1):52–60.

Kaandorp, G. C. and Koole, G. (2007). Optimal outpatient appointment scheduling. *Health Care Management Science*, 10:217–229.

Men, C. (2009). *Optimization models for radiation therapy: treatment planning and patient scheduling*. University of Florida.

Mulholland, M. W., Abrahamse, P., and Bahl, V. (2005). Linear programming to optimize performance in a department of surgery. *Journal of the American College of Surgeons*, 200(6):861–868.

Pham, D. N. and Klinkert, A. (2008). Surgical case scheduling as a generalized job shop scheduling problem. *European Journal of Operation Research*, 185:1011–1025.

Robbins, W. A. and Tuntiwongpiboon, N. (1989). Linear programming a useful tool in case-mix management. *Healthcare Financial Management*, 43(6):114–116.

Vanberkel, P. T., Boucherie, R. J., Hans, E. W., and Hurink, J. L. (2011). Optimizing the strategic patient mix. Memorandum 1935, Department of Applied Mathematics, University of Twente, Enschede.

Vissers, J., Adan, I. J., and Bekkers, J. A. (2005). Patient mix optimization in tactical cardiothoracic surgery planning: a case study. *IMA Journal of Management Mathematics*, 16(3):281–304.

3. PROTON THERAPY PATIENT SCHEDULING: MARKOV DECISION PROCESS MODELING APPROACH

Ridvan Gedik Chase Rainwater Shengfan Zhang

3.1 Introduction

Stochastic patient arrivals and appointment cancellations (no-shows) are the most common sources of uncertainty in patient scheduling problems. However, in Chapter 2, we assume that the issue of rescheduling and treatment session cancellations is less significant in proton therapy treatment planning than in other healthcare applications due to the large demand for this therapy. Therefore, we study deterministic patient scheduling models in bi-criteria objective settings. As the costs associated with building proton therapy facility decrease due to advancing technology, handling stochastic patient arrivals will be vital in the near future to keep the utilization of the resources at their maximum levels. Therefore, we propose a Markov decision process (MDP) model that accounts for stochastic patient arrivals to a proton therapy facility.

Patient mix optimization, the requirement that the mix of patients treated satisfy desired percentages, is a relatively new consideration in proton therapy planning. The desirable findings in hospitals in early stages of DRG (i.e. case mix) implementations by LP models motivate further investigation into sophisticated models for efficient patient planning in the proton therapy healthcare environment. All other studies in the patient mix optimization literature aim to find the most suitable patient mix levels such that the maximum benefit is achieved. On the other hand, none of them focused on patient scheduling where the patient mix levels are taken as an input and the impacts of their interactions with the other operational constraints are demonstrated on the facility capacity. In Chapter 2, we demonstrated the impacts of a given patient mix preferences on the capacity of a proton therapy facility by comparing the total treated patients versus the total deviations from patient mix levels in the presence of several operational constraints. The tradeoffs between total fractions delivered and total deviation from the mix ratios were identified through analysis of efficient frontiers.

Assuming that there is an infinite patient supply for the treatment is a simplifying assumption that will become less appropriate as proton therapy becomes an affordable treatment. To address this concern, this chapter proposes an MDP model which relaxes the assumption that there is an infinite patient supply for treatment. Instead, this model is able to represent the stochastic arrivals of patients which also allows incorporating other stochastic dynamics, such as appointment cancellations that happen on daily basis. Even though MDP is a powerful way of handling stochasticity associated with patient arrivals, it takes quite amount of computational effort to solve this multi-category patient scheduling problem due to the exponentially growing state space. Therefore, this chapter also proposes an approximate aggregate MDP model to overcome this special type of *curse of dimensionality*. The aggregate MDP model clusters original states into subgroups that results in a more tractable state space. Of course, the aggregate MDP provides only approximate optimal patient admission policies. Therefore, we seek to assess the trade-off between the traditional and aggregated models in terms of solution time and quality.

The rest of this chapter is organized as follows. Section 3.2 describes how stochastic patient arrival and scheduling in healthcare facilities have been handled by MDP models in the literature. Then, two MDP models, exact and aggregate, are introduced in Section 3.3. Section 3.4 demonstrates the performance of these models and finally, Section 3.5 briefly highlights the contributions of this study and discusses future research directions.

3.2 Literature Review

Adan and Vissers (2002) categorize patient admissions into two groups; non-scheduled and scheduled. Non-scheduled admissions are unplanned and might be due to emergency cases, whereas scheduled patients are planned and selected from a waiting list created as the service is being requested. In most of the studies (e.g., Bowers and Mould (2005), Cayirli et al. (2006), Pham and Klinkert (2008)), scheduled (elective) cases are considered in order to reduce potential uncertainties associated with patient attributes (i.e., patient type, financial gain, resource allocation per patient, etc.). Some other studies assume that the treatment time for an outpatient is constant for the

sake of simplicity (see, e.g., Conforti et al. (2008)). In the context of this study, we assume that proton therapy patients are elective outpatients who do not need to spend the night at the facility.

Conforti et al. (2010) compare the two most common types of daily basis radiation therapy patient scheduling strategies: (i) block system and (ii) non-block system. In blocked scheduling, a workday is composed of a fixed number of blocks/time slots with the same duration, whereas in the non-block scheduling, different time intervals are reserved for different patients based on the type of the treatments. Furthermore, patients in the non-block scheduling system are usually scheduled on a first-come first-served basis unless the individual cases are not restricted by strict earliest and/or latest start constraints. In the same study, after highlighting the fact that the block scheduling strategy is more commonly used than its counterpart in many radiation therapy centers, a patient scheduling model which accounts for non-block scheduling rules is developed for a short planning period (a week). In reality, non-block appointment system is conceptually superior to the block appointment system since the former is able to represent the whole workload. This is because, during a workday, accumulation of the idle time leftover from a time slot can be used to treat other patients who have not been scheduled yet.

Despite of its advantages, the non-block scheduling appointment strategy is not considered very often in the literature due to the extra limitations/constraints it adds to the treatment planning problem. On the other hand, block scheduling is commonly encountered (Conforti et al., 2008; Kapamara et al., 2006; Gocgun et al., 2011; Nunes et al., 2009), since it simplifies most of the concepts that are hard to be taken into account in a radiation therapy patient scheduling problem. Therefore, due to its ability to simplify appointment scheduling decisions, the proton therapy patient scheduling problem *with stochastic arrivals* is assumed to have a block appointment system. In such system, a group of patients can be assigned to time slots in a manner such that each of them can concurrently be receiving treatment. Figure 3.1 demonstrates a basic block patient scheduling schema for a configuration of three gantries and three time slots in a given day. Hence, for each time slot, the number of patients assigned to a time slot can not be greater than the number of gantries available in that time slot in the system. Accordingly, let us assume that

the days in the planning period are divided into time slots such that all these slots have identical durations. Note that the daily treatment times of all gantries are identically distributed to time slots of a day as demonstrated in Figure 3.1. Therefore, the total number of patients that can be treated on all gantries in a day would be equivalent to the total number of time slots allocated to all gantries in a given day (i.e. 9 in Figure 3.1).

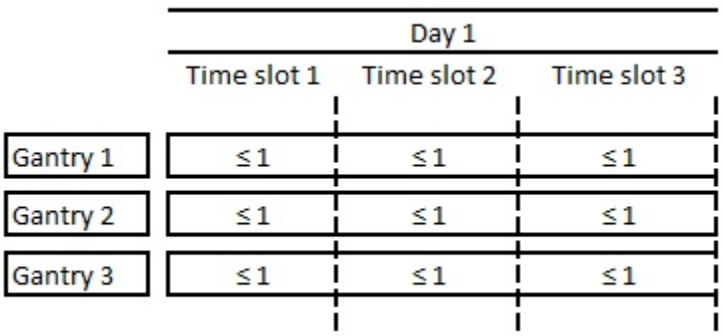

Figure 3.1: Block appointment system

A recent contribution that is directly related to proton therapy patient scheduling research corresponds to the work of Nunes et al. (2009). They present an MDP model in order to control the scheduled patient admissions from different specialties on a periodic basis. Patient flows from *m* distinct specialties are assumed to be continuous. In order to measure the resource consumption, they adopt the *treatment pattern* methodology, which is first proposed by Kapadia et al. (1985). It is assumed that a patient could follow *n* different treatment patterns until he is discharged (end of treatment). In this study, state space of the hospital is defined as the number of patients from all specialties following different treatment patterns. The number of patients to be admitted to the hospital in each specialty for the next decision period is considered as the action space. Finally, the stochastic dynamics of the study is described as the probability of one patient's transition from a treatment pattern to all others.

One of the first MDP approaches developed for hospital admission scheduling is proposed by Kolesar (1970) who transforms the MDP model into a linear program to obtain results. Another important finding of this study is that simultaneous patient scheduling reservations for dif-

ferent services are modeled over a planning horizon.

Gocgun et al. (2011) develop a finite horizon MDP to model a patient scheduling problem commonly seen in providing computed tomography (CT) service. They assume that there are only four patient types who demand the CT service during a work day period. The MDP model in this study aims to maximize the total profit obtained by scheduling different patient types for the available time slot of a CT machine during the work day. Different than most of the patient scheduling models, they adopt each available time slot for each CT machine as a *stage*. Similar to the study conducted by Nunes et al. (2009), the number of patients from different types constitutes the state space and the actions are denoted by the number of scheduled patients from different types for the next service slot. They compare the optimal solutions obtained from their model with simple heuristic rules (first-come first-served (FCFS), randomized etc.) typically employed to schedule patients in real life. Even though their model finds better values in terms of total profit, FCFS heuristic provides better values in terms of the average number of patients not scanned by the end of the day.

3.3 Solution Methodology

3.3.1 A Markov Decision Process (MDP) Model

In light of the insights developed by Nunes et al. (2009); Kapadia et al. (1985); Gocgun et al. (2011), we also model our strategic patient mix optimization model as an MDP. Based on the block scheduling schema in Figure 3.1, the finite state and action space for of our problem can be modeled as in equations (3.1) and (3.2). s_{tq}^{k} represents the number of patients being treated from category k in service slot q between decision instants $t-1$ and t. Thus, the state space S comprises all possible states. Furthermore, a_{q}^{k} represents the number of patients in each category that will be admitted for treatment in service slot q in the next period.

$$S = \begin{pmatrix} s^1_{t,q=1} \\ s^2_{t,q=1} \\ \vdots \\ s^{|K|}_{t,q=1} \end{pmatrix}, \begin{pmatrix} s^1_{t,q=2} \\ s^2_{t,q=2} \\ \vdots \\ s^{|K|}_{t,q=2} \end{pmatrix}, \ldots, \begin{pmatrix} s^1_{t,q=|Q|} \\ s^2_{t,q=|Q|} \\ \vdots \\ s^{|K|}_{t,q=|Q|} \end{pmatrix} \tag{3.1}$$

$$A_{s_t} = \begin{pmatrix} a^1_{q=1} \\ a^2_{q=1} \\ \vdots \\ a^{|K|}_{q=1} \end{pmatrix}, \begin{pmatrix} a^1_{q=2} \\ a^2_{q=2} \\ \vdots \\ a^{|K|}_{q=2} \end{pmatrix}, \ldots, \begin{pmatrix} a^1_{q=|Q|} \\ a^2_{q=|Q|} \\ \vdots \\ a^{|K|}_{q=|Q|} \end{pmatrix} \tag{3.2}$$

In addition to the number of current patients receiving treatment and newly accepted patients from each category, we also need to account for the ones whose treatment schedule ends on each day. From each patient type k on day t, it is known that there will be $a^k_{t-n_k}$ number of discharges, which is the number of patients from type k accepted on day $t - n_k$ given that a patient from category k must receive fractions during n_k consecutive days. It should be noted that incorporating the discharged patients into the state definitions would increase the complexity and degrade the tractability of the model. Therefore, a useful assumption is made in order to avoid this foreseen difficulty. Accordingly, discharges are assumed to be handled after accepting new patients to each category. Hence, transition from one state to another includes only the newly accepted patients and the ones that are receiving their treatments.

In the case of identical gantries, we can modify the gantry-time slot representation of a single day as in Figure 3.1. Accordingly, state space and action space can be defined in a more compact way as in equations (3.3) and (3.4). Based on the representation in Figure 3.1, both time slots and the gantries are assumed to be identical in a given day. Moreover, during our work in Chapter 2 on the patient scheduling problem, we have identified that if the daily gantry capacity depends only on gantry type not time period, an optimal solution for the linear program exists in which we start and treat the same number of patients in category k on each day t and in each

gantry g. In other words, this implies that the optimal solution of the LP model *enforces patient types to hold the desired patient mix restrictions on daily basis*. In order to represent this, s^k is defined as the total number of type k patients receiving treatments in all time slots and a^k is defined as the accepted number of patients from category k to all time slots.

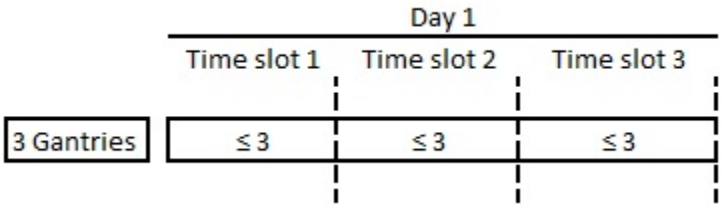

Figure 3.1: Modified block appointment system for identical gantries

$$S = \begin{pmatrix} s^1_{q=1} + \ldots + s^1_{q=|Q|} \\ s^2_{q=1} + \ldots + s^2_{q=|Q|} \\ \vdots \\ s^{|K|}_{q=1} + \ldots + s^{|K|}_{q=|Q|} \end{pmatrix} = \begin{pmatrix} s^1 \\ s^2 \\ \vdots \\ s^{|K|} \end{pmatrix} \tag{3.3}$$

$$A_s = \begin{pmatrix} a^1_{q=1} + \ldots + a^1_{q=|Q|} \\ a^2_{q=1} + \ldots + a^2_{q=|Q|} \\ \vdots \\ a^{|K|}_{q=1} + \ldots + a^{|K|}_{q=|Q|} \end{pmatrix} = \begin{pmatrix} a^1 \\ a^2 \\ \vdots \\ a^{|K|} \end{pmatrix} \tag{3.4}$$

Due to the problem definition, patient mix ratios must be satisfied and maximum number of fractions/patients must be treated over a planning period. In order to assess the impacts of these constraints, we represent patient mix preferences in the optimality equations. One procedure is to assign a penalty for those patient types which fail to conform to the desired patient mix levels. To implement this procedure, we must keep track of the total number of patients treated (after the actions are taken), and then compare both the actual and desired patient mix levels. Let $\mathbb{J}$ be the

total number of patients from all types being treated in the facility.

$$\mathbb{J} = \sum_{k \in K} \{s^k + a^k\} \tag{3.5}$$

Then, the desired (D_k) and actual (H_k) patient mix of patient type k can be calculated as follows.

$$D_k = d_k \mathbb{J} \tag{3.6}$$

$$H_k = s^k + a^k \tag{3.7}$$

Hence, deviation from patient mix constraint for each category k is $|A_k - D_k|$. Then, a reward function ($r(s,a)$) for each state and action can be constructed as follows:

$$r(s,a) = \sum_{k \in K} a^k n_k f_k - \sum_{k \in K} (w_k |H_k - D_k|) \qquad a \in A_s \tag{3.8}$$

where f_k is the number of fractions required by patient type k per day and n_k is the consecutive number of days a patient type k needs to receive treatment. Therefore, the first term in the reward function is the total number of fractions obtained by starting a^k patients from each patient type $k \in K$. The second term penalizes the deviation of each type k from the desired patient mix level by w_k. It is important to note that the deviation is measured in terms of number of patients, but the first term is in terms of fractions. Finally, it is clear to see that the objective seeks to maximize the total reward.

Let $v(s)$ be the total expected reward obtained for the state s. Assuming that the state and action spaces are finite and the horizon length is infinite, a value iteration algorithm finds a stationary ε-optimal policy and approximation to value function (3.9) for each state s with a discount factor of β.

$$v(s) = \max_{a \in A_s} \{r(s,a) + \beta \sum_{j \in S} p_{js} v(j)\} \tag{3.9}$$

We note that p_{js} is the transition probability from state j to s. The stochastic dynamic of the problem, patient arrivals, can be modeled as Poisson arrivals. Hence, let $N(k)$ be the number of patient types k request treatment on a specific day in the proton therapy facility. Then, $N(k) \sim Poisson(\alpha_k)$. Assuming that the arrivals are independent from each other, we can estimate the probability of patient arrivals as follows:

$$P(N(1)=x, N(2)=y, \ldots, N(|K|)=z) = \frac{e^{-\alpha_1}\alpha_1^x}{x!}\frac{e^{-\alpha_2}\alpha_2^y}{y!}, \ldots, \frac{e^{-\alpha_{|K|}}\alpha_{|K|}^z}{z!} \tag{3.10}$$

A Numerical Example

This section demonstrates the performance of the MDP model over a numerical example in terms of best patient admission policy and convergence graphs for each state. Moreover, since the best patient admission policies may not always be optimal due to some other operational restrictions, second and third best actions are also reported. Before explaining these findings, the transition probability matrix, action and state space are illustrated in Figure 3.2 and 3.3. For this illustration the total patient capacity (Q) is equal to 4.

States	Actions
(0,0)	[(0,0), (0,1), (1,0), (0,2), (1,1), (2,0), (0,3), (1,2), (2,1), (3,0),(>4)]
(0,1)	[(0,0), (0,1), (1,0), (0,2), (1,1), (2,0), (>3)]
(1,0)	[(0,0), (0,1), (1,0), (0,2), (1,1), (2,0), (>3)]
(0,2)	[(0,0), (0,1), (1,0), (>2)]
(1,1)	[(0,0), (0,1), (1,0), (>2)]
(2,0)	[(0,0), (0,1), (1,0), (>2)]
(0,3)	[(0,0), (>1)]
(1,2)	[(0,0), (>1)]
(2,1)	[(0,0), (>1)]
(3,0)	[(0,0), (>1)]
Full	[(>0)]

Figure 3.2: State space and action space when $Q = 4$

The reason behind the limited and decreasing order action space as the number of patients in the facility increases is due to the assumption that discharges are performed after accepting

Probability Transition Matrix											
	(0,0)	(0,1)	(1,0)	(0,2)	(1,1)	(2,0)	(0,3)	(1,2)	(2,1)	(3,0)	Full
(0,0)	0.41	0.16	0.2	0.03	0.08	0.05	0	0.02	0.02	0.01	0.01
(0,1)	0	0.41	0	0.16	0.2	0	0.03	0.08	0.05	0	0.06
(1,0)	0	0	0.41	0	0.16	0.2	0	0.03	0.08	0.05	0.06
(0,2)	0	0	0	0.41	0	0	0.16	0.2	0	0	0.23
(1,1)	0	0	0	0	0.41	0	0	0.16	0.2	0	0.23
(2,0)	0	0	0	0	0	0.41	0	0	0.16	0.2	0.23
(0,3)	0	0	0	0	0	0	0.41	0	0	0	0.59
(1,2)	0	0	0	0	0	0	0	0.41	0	0	0.59
(2,1)	0	0	0	0	0	0	0	0	0.41	0	0.59
(3,0)	0	0	0	0	0	0	0	0	0	0.41	0.59
Full	0	0	0	0	0	0	0	0	0	0	1

Figure 3.3: Probability transition matrix when $Q = 4$

new patients. When the capacity (the absorbing "FULL" state) is reached, we assume that the MDP model is terminated. First column in Figure 3.2 lists all possible states for this problem. Note that first number in each state represents the number of patients in the facility from the first category and the second number stands for the number of patients in the facility from the second category. Actions are the possible combinations of accepting patients to the facility from these two types.The last action of each state in Figure 3.2 groups all combinations that might lead to the full state under a single action. Similarly, the probability of reaching the full state is calculated by subtracting the probabilities of all other arrival combinations from 1.

Table 3.1: Input parameters

	n_k	f_k	w_k	d_k	α_k
k=1	40	1	40	50%	0.5
k=2	30	1	30	50%	0.4

Baseline input parameters are seen in Table 3.1. Based on the desired mix percentages (d_k), the number of patients from two types are assumed to be the same for simplicity of this example. The number of consecutive days required for type k over the planning period (n_k) and number of daily fractions required for type k (f_k) are the actual numbers of the two categories obtained from University of Florida Proton Therapy Institute (UFPTI). Arrival rates (α_k) and penalty terms (w_k) are selected arbitrarily.

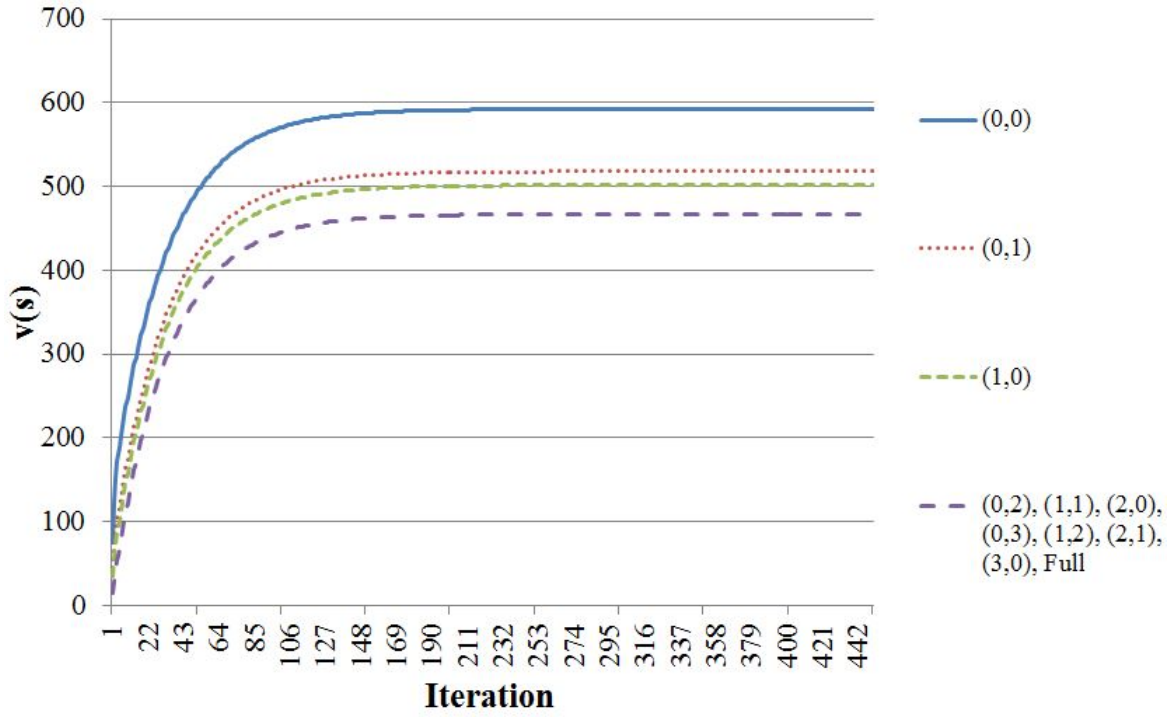

Figure 3.4: $v(s)$ when $w_1 = 40, w_2 = 30$

Figure 3.4 demonstrates the convergence of value function for each state obtained by solving the optimality equations 3.9 with the input parameters in Table 3.1 via a value iteration algorithm. Similarly, Figure 3.5 and 3.6 illustrate the change in these functions with different penalty terms.

These figures depict that the value function of the state (0,0) dominates the value function of all other states. This is because, the fewer the number of patients in the facility, the more spots that can be filled with new patients which ultimately leads to a larger total expected fractions. As the penalty parameters decrease, the actions associated with the full state become less desirable for the states. The model chooses to transition to the full state instead of deviating from desired mix levels as the penalty terms increase. For this reason, transitioning to the full state is the best action for the last seven, six and five states as seen in Figure 3.4, 3.5 and 3.6, respectively.

Since the number of accepted patients directly impacts the actual and desired total mix ratios, acceptance preferences (best, second and third best actions) change as the penalty terms differ. This is illustrated for each state in Table 3.2. For instance, the best action is to accept two and

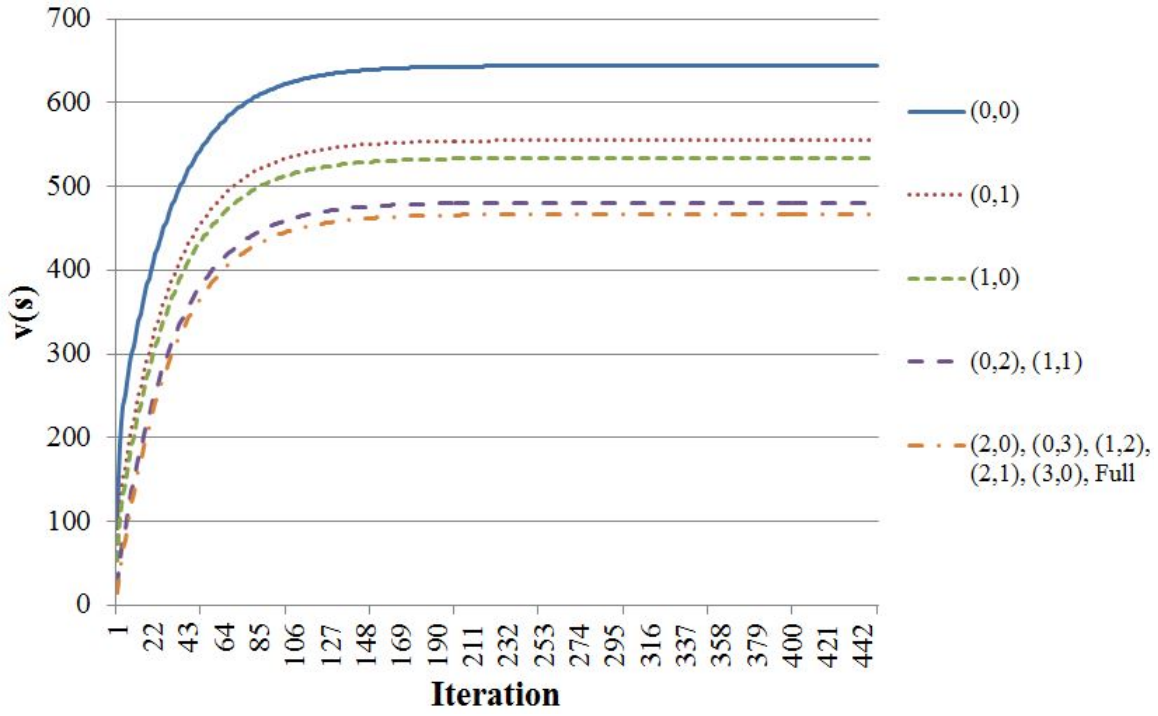

Figure 3.5: $v(s)$ when $w_1 = 20, w_2 = 15$

one patient from the first and second category, respectively, for the state (0,0) regardless of the penalty terms. On the other hand, accepting one patient from the first category becomes more affordable and the best action as the penalty terms decrease for the state (0,2). Similar conclusions can be made for each state.

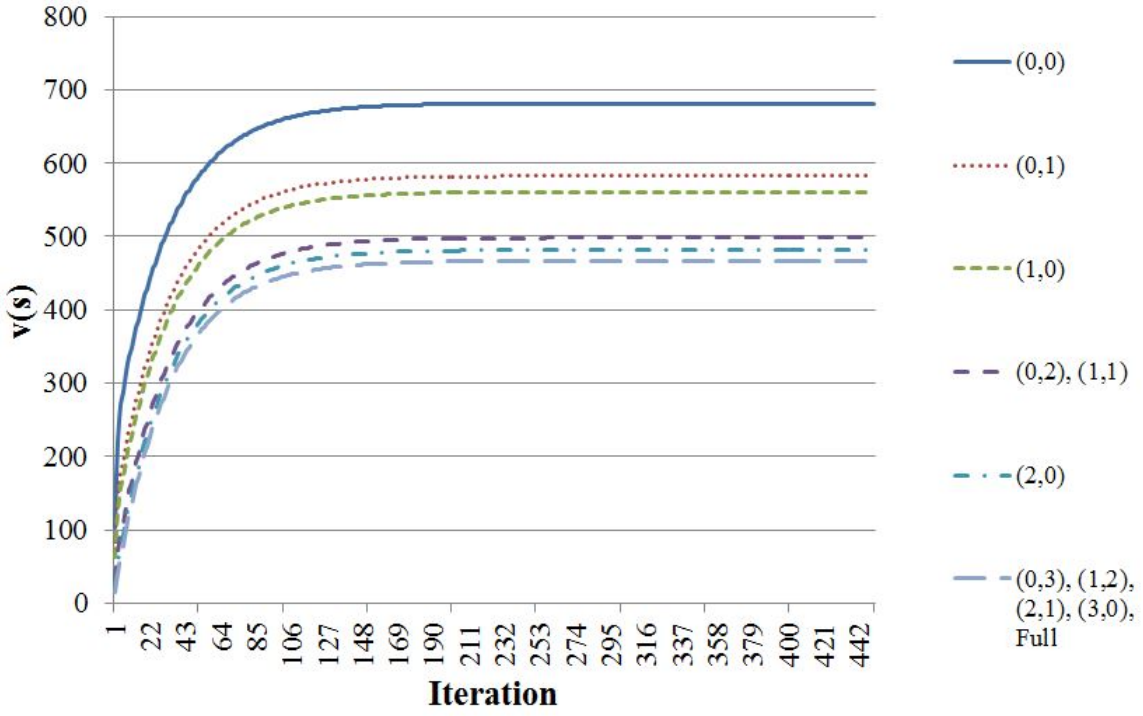

Figure 3.6: $v(s)$ when $w_1 = 8, w_2 = 6$

Table 3.2: Best, second best and third best actions

		State										
Penalty	Action	(0,0)	(0,1)	(1,0)	(0,2)	(1,1)	(2,0)	(0,3)	(1,2)	(2,1)	(3,0)	Full
$w_1 = 40$, $w_2 = 30$	Best	(2,1)	(2,0)	(1,1)	> 2	> 2	> 2	> 1	> 1	> 1	> 1	> 0
	Second Best	(1,1)	(1,0)	(0,1)	(1,0)	(1,0)	(0,1)	(0,0)	(0,0)	(0,0)	(0,0)	>0
	Third Best	(1,2)	(1,1)	(0,2)	(0,0)	(0,0)	(1,0)	(0,0)	(0,0)	(0,0)	(0,0)	>0
$w_1 = 20$, $w_2 = 15$	Best	(2,1)	(2,0)	(1,1)	(1,0)	(1,0)	>2	>1	>1	>1	>1	>0
	Second Best	(1,2)	(1,1)	(0,2)	>2	>2	(0,1)	(0,0)	(0,0)	(0,0)	(0,0)	>0
	Third Best	(1,1)	(1,0)	(0,1)	(0,1)	(0,1)	(1,0)	(0,0)	(0,0)	(0,0)	(0,0)	>0
$w_1 = 8$, $w_2 = 6$	Best	(2,1)	(2,0)	(1,1)	(1,0)	(1,0)	(0,1)	>1	>1	>1	>1	>0
	Second Best	(3,0)	(1,1)	(2,0)	>2	(0,1)	(1,0)	(0,0)	(0,0)	(0,0)	(0,0)	>0
	Third Best	(1,2)	(1,0)	(0,2)	(0,1)	>2	>2	(0,0)	(0,0)	(0,0)	(0,0)	>0

3.3.2 An Aggregate MDP Model

The MDP model in the previous section provides strategic level optimal patient admission policies when the time horizon is assumed to be in steady state. Despite the useful practical insights it provides for the decision makers, it becomes intractable as the problem parameters increase. The state space of the MDP model grows exponentially in the order of $|Q|$ and $|K|$ (see Section 3.4).Thus, this section introduces a state aggregation technique that is proposed to overcome the exponentially growing states and actions (*curse of dimensionality*) associated with the MDP formulation.

In order to reduce the complexity associated with a large state space, researchers have focused on applying aggregation techniques in which the main idea is clustering original states into aggregate subgroups. These aggregate subgroups are treated as newly created states in the aggregate MDP. As a consequence, the reduced number of states will directly decrease the size of the probability transition matrix since it is the square of the number of states. Using this approach, the original MDP can be represented by a potentially exponentially smaller *approximate* MDP. Since the ultimate approximate model preserves the Markovian property, any MDP algorithm can be used to solve the *aggregate* MDP.

State aggregation is referred to as state abstraction in the studies within the artificial intelligence domain. Dearden and Boutilier (1997) provide a very concise discussion regarding the differences between state abstraction (or aggregation) and other compact MDP representation techniques. They propose an MDP model for the decision-theoretic planning (DTP) problem, another version of optimal stochastic control problem, and develop an abstract MDP model with exponentially fewer states which allow them to obtain approximate optimal solutions. The key idea behind their abstraction policy is to use action and reward structure to judge whether a variable (state) should be included in the aggregate MDP or not. The selection criteria of a state is based on the eligibility of capturing important behaviors of the original MDP (i.e. set of possible actions, amount of reward). Although there may be several states removed from the original MDP, the ultimate policy from their aggregate MDP remains executable for the original MDP. In other

words, the optimal policy for the aggregate problem is the approximate optimal solution for the original problem. This is a very important feature of a good practical state aggregation algorithm and one of the main objectives of this study in generating approximate optimal patient admission policies.

Dean et al. (1997) introduce a ε-*homogeneity* technique that is employed to partition the very large state space of MDPs. They specifically study approximate optimal solutions for bounded MDPs in which upper and lower bounds on the transition probabilities and rewards are given as input parameters. Li et al. (2006) propose a unified treatment of state abstraction for MDPs with large state space by analyzing five different state aggregation techniques and assess their functionalities in planning and learning problems. Moreover, an insightful case study is offered to measure the tradeoff between minimizing curse of dimensionality through different state abstraction and minimizing information loss. Other applications of state abstraction techniques in reinforcement learning can be found in Dietterich (2000); Sutton et al. (1999) and Andre and Russell (2002).

Despite its popularity and dominance in artificial intelligence field, especially reinforcement learning application area, state aggregation has a very strong competitor in operation research field. Researchers in operations research have recently started exploring a different version of tackling curse of dimensionality through approximate dynamic programming (ADP) techniques. As opposed to clustering states in state aggregation, ADP focuses on stepping forward in time and selecting states with good approximate dynamic value functions to obtain approximately optimal policies (Powell, 2009). This technique has been successfully implemented on numerous challenging problems including multidimensional knapsack (Bertsimas and Demir, 2002), transportation & logistics (Powell et al., 2012), resource allocation (Powell et al., 2003) and capacity allocation (Schütz and Kolisch, 2012; Sauré et al., 2012). For further details on different aspects of ADP techniques, we refer the reader to Powell (2007).

3.3.3 State Aggregation

Recall that $S = \{s^1, s^2, \ldots, s^{|K|}\}$ is the state space of the original MDP model where s^k is defined as the total number of type k patients within the facility. We let $\{S_1, S_2, \ldots, S_{|K|}, S_{|K|+1}, S_{|K|+2}\}$ partition S where $\cup_{k=1}^{|K|+2} S_k = S$ and $S_k \cap S_m = \emptyset$ if $k \neq m$ and $k, m \in \mathbb{K} = \{1, 2, \ldots, |K|, |K|+1, |K|+2\}$. $\mathbb{K}$ is the set of states in the aggregate MDP. $S_{|K|+2}$ represents the full state in the aggregate MDP. If $s^1 = s^2 = \ldots = s^{|K|}$ holds, that is the number of patients from all patients are same, in an original MDP state, then it is assigned to the $S_{|K|+1}$ aggregate state. We assign an original MDP state $\{s^1, s^2, \ldots, s^{|K|}\}$ to aggregate state S_k if $s^k > s^m$ for all $m \in K$ and $m \neq k$ (patient type k has the unique largest number of patients). If there is more than one patient types which have the identical maximum amount of patients, then we randomly pick the aggregate state among those states. For example, if $s^k = s^m > s^n$ for all $n \in K$ and $n \neq k \neq m$ in $\{s^1, s^2, \ldots, s^{|K|}\}$, then we randomly assign this original state to either S_k or S_m with equal probability.

For each $k \in \mathbb{K}$, we define $\mathbb{A}_k$ as $\cup_{s \in S_k} A_s$. That is, $\mathbb{A}_k$ is the set of actions available from state $k \in \mathbb{K}$ in the aggregate MDP. We use the *fixed-weight aggregation* technique that is proposed by Heyman and Sobel (2003). Accordingly, for each $k, m \in \mathbb{K}$ and $a \in \mathbb{A}_k$, we let γ_{km} and ρ_k^a be a transition probability and single-state immediate reward in the aggregate MDP as defined in equations (3.11) and (3.12), respectively,

$$\gamma_{km} = \sum_{s \in S_k} \lambda_s^k \sum_{j \in S_m} p_{sj} \tag{3.11}$$

$$\rho_k^a = \sum_{s \in S_k} \lambda_s^k r(s, a) \tag{3.12}$$

where $\lambda_s^k \geq 0$ and $\sum_{s \in S_k} \lambda_s^k = 1$ for each $k \in \mathbb{K}$ and $s \in S_k$. In (3.11), $\sum_{j \in S_m} p_{sj}$ is the aggregate probability of transitioning from state s to aggregate state S_m which makes γ_{km} the weighted average of these transition probabilities where λ_s^k are the weights for $s \in S_k$. Intuitively, λ_s^k assesses the contribution of state s to the aggregate state S_k and can be any valid weighting function (Li

et al., 2006). It is easy to see that γ_{km} are the transition probabilities in the aggregate MDP:

$$\sum_{m\in\mathbb{K}} \gamma_{km} = \sum_{m\in\mathbb{K}} \sum_{s\in S_k} \lambda_s^k \sum_{j\in S_m} p_{sj} = \sum_{s\in S_k} \lambda_s^k \sum_{m\in\mathbb{K}} \sum_{j\in S_m} p_{sj} = \sum_{s\in S_k} \lambda_s^k \sum_{j\in S} p_{sj} = 1$$

Similarly, ρ_k^a is the weighted average of $r(s,a)$ as defined in (3.8).

3.3.4 Fixed Policy Evaluation

Heyman and Sobel (2003) prove that the output of the fixed aggregation transformation technique is indeed an MDP model with fewer states. This enables us to use the same value iteration algorithm to solve the aggregate model. The policy obtained by solving the aggregate model is executable for the original MDP model since the set of aggregate actions for each aggregate group is inherited from the original states ($\mathbb{A}_k$=$\cup_{s\in S_k} A_s$). Of course, solutions for both original and aggregate MDP may not be identical. Thus, we need a fixed policy evaluation technique to assess the quality of the aggregate MDP policy with respect to the optimal policy. In other words, a disaggregating method is required to substitute the policy for the aggregate MDP model into the original MDP.

The optimal policy for infinite horizon MDP model is always stationary. That is, the optimal stationary policy (π^*) for an infinite horizon discounted MDP model does not change with time and remains identical once it is obtained. Given a policy π, we define $V_\pi(s)$ as the value of policy π at state s, $R_\pi(s)$ as the immediate reward and $P_\pi(s,j)$ as the probability transition matrix. Let V_π and R_π be n dimensional column vectors and P be an $n \times n$ matrix. Note that for any given policy π, R_π and P will be known which will make the solution of the following set of linear equations (3.13) possible with respect to only unknown V_π.

$$\begin{aligned} V_\pi &= R_\pi + \beta P_\pi V_\pi \\ R_\pi &= (I - \beta P) V_\pi \\ V_\pi &= (I - \beta P)^{-1} R_\pi \end{aligned} \tag{3.13}$$

After solving the aggregate MDP model by the value iteration algorithm, we obtain the approximate optimal policy $\bar{\pi}$. We create another feasible policy δ for the original MDP model based on $\bar{\pi}$ such that

$$\delta(s) = a \text{ if } s \in S_k \text{ and } \bar{\pi}(k) = a \; \forall k \in \mathbb{K}. \tag{3.14}$$

The policy disaggregation (3.14) suggests using the same optimal action for the aggregate subgroup S_k for all the original states assigned to it ($s \in S_k$). The quality of the aggregate policy δ and optimal policy π^* is measured by solving the set of linear equations (3.13).

3.4 Computational Results

3.4.1 Problem Parameters and Experimental Design

Several different problem instances are used to test the tractability and efficiency of the aggregate MDP model. Table 3.1 illustrates the required parameters for each patient category. All of these parameters except daily arrival rates are obtained from UFPTI. The arrival rate is calculated by using expected number of new incidence of each patient category given in American Cancer Society (2014). These values are first adjusted based on the population of Florida and then normalized to fit into $[0,1]$. Finally, the arrival rate of each patient is obtained by multiplying the normalized value by four to make the total arrival rate of the patients equal to the minimum number of time slots available in the facility.

Different combinations of the number of patient categories ($|K| = 5,6,\ldots,10$) and available time slots ($Q = 4,5,6$) are used to vary the size of the state space. Therefore, we group multiple patient types as shown in Table 3.2-3.6 to meet different number of patient categories in problem instances with $|K| = 5,6,\ldots,9$, respectively. In addition, three different penalty terms are adopted as follows:

1. $w_k = d_k \times 20 \; \forall k \in K$

2. $w_k = \frac{n_k \times f_k}{4} \; \forall k \in K$

3. $w_k = n_k \times f_k \times d_k \; \forall k \in K$

Lastly, state aggregation weights are calculated as in equation (3.15).

$$\lambda_s^k = \frac{1}{|S_k|} \quad \forall k \in \mathbb{K},\, s \in S_k \tag{3.15}$$

Table 3.1: (Aggregate) MDP problem parameters when $|K| = 10$

	Patient Category ($k \in K$)									
	1	2	3	4	5	6	7	8	9	10
d_k	65%	15%	7%	3%	3%	2%	1%	1%	2%	1%
n_k	40	40	31	31	30	30	30	30	42	12
f_k	1	1	2	2	1	1	1	1	1	1
α_k	1.65	0.38	0.92	0.19	0.12	0.08	0.15	0.15	0.19	0.15

Table 3.2: (Aggregate) MDP problem parameters when $|K| = 9$

	Patient Category ($k \in K$)								
	1	2	3 & 10	4	5	6	7	8	9
d_k	65%	15%	8%	3%	3%	2%	1%	1%	2%
n_k	40	40	31	31	30	30	30	30	42
f_k	1	1	2	2	1	1	1	1	1
α_k	1.65	0.38	1.07	0.19	0.12	0.08	0.15	0.15	0.19

Table 3.3: (Aggregate) MDP problem parameters when $|K| = 8$

	Patient Category ($k \in K$)							
	1	2	3 & 10	4	5	6	7 & 8	9
d_k	65%	15%	8%	3%	3%	2%	2%	2%
n_k	40	40	31	31	30	30	30	42
f_k	1	1	2	2	1	1	1	1
α_k	1.65	0.38	1.07	0.19	0.12	0.08	0.30	0.19

Table 3.4: (Aggregate) MDP problem parameters when $|K| = 7$

	Patient Category ($k \in K$)						
	1	2	3 & 10	4	5 & 6	7 & 8	9
d_k	65%	15%	8%	3%	5%	2%	2%
n_k	40	40	31	31	30	30	42
f_k	1	1	2	2	1	1	1
α_k	1.65	0.38	1.07	0.19	0.21	0.30	0.19

Table 3.5: (Aggregate) MDP problem parameters when $|K| = 6$

	Patient Category ($k \in K$)					
	1	2	3 & 10	4	5,6,7 & 8	9
d_k	65%	15%	8%	3%	7%	2%
n_k	40	40	31	31	30	42
f_k	1	1	2	2	1	1
α_k	1.65	0.38	1.07	0.19	0.51	0.19

Table 3.6: (Aggregate) MDP problem parameters when $|K| = 5$

	Patient Category ($k \in K$)				
	1 & 2	3 & 10	4	5, 6,7 & 8	9
d_k	80%	8%	3%	7%	2%
n_k	40	31	31	30	42
f_k	1	2	2	1	1
a_k	2.03	1.07	0.19	0.51	0.19

3.4.2 MDP vs. Aggregate MDP Model

The patient scheduling problem formulations in Section 3.3 are modeled in JAVA programming language. We assess the quality of the aggregate policy (δ) and optimal policy (π^*) as described in Section 3.3.4. Recall that these policies provide the best action for each state and the total expected rewards obtained for each state are calculated by solving (3.13). In order to compare the values and best actions obtained by the MDP and aggregate MDP model, we define two performance indicators:

$$\text{Average percent difference (APD)} = 100 \times \frac{\sum_{s \in S} \left(\frac{|V_{\pi^*}(s) - V_{\delta}(s)|}{\max\{V_{\pi^*}(s) - V_{\delta}(s)\}} \right)}{|S|} \tag{3.16}$$

$$\text{Matched action percentage (MAP)} = 100 \times \frac{\text{Number of same actions across all states in } S}{\text{Total number of states } (|S|)} \tag{3.17}$$

APD demonstrates how much, on average, the total expected reward values obtained by the MDP and aggregate MDP model deviate from each other across all states, whereas MAP illustrates the average precision of the aggregate MDP model in terms of locating the same actions with the MDP model across all states. These two measures for the instances with penalty type 1, 2 and 3 are illustrated in Figures 3.1-3.3, Figures 3.4-3.6 and Figures 3.7-3.9, respectively. When the penalty type 2 is used, that is each unit of deviation from the desired patient mix level for category k is penalized by $\frac{n_k f_k}{4}$, APD and MAP levels are consistently decreasing and increasing, respectively, as the models are exposed to larger $|K|$ values. This leads us to conclude that behaviors of the MDP and aggregate MDP models become similar as $|K|$, therefore, the number of states increases when the penalty 2 levels are used. We can see the same pattern in APD and MAP values when each unit deviation from the desired mix level for patient type k is charged with $n_k f_k d_k$ (penalty type 3). The only instance that does not comply with this trend under penalty type 3 is when $|K| = 6$ and $Q = 6$. Even though the APD and MAP values improve with larger $|K|$ when penalty type 1 is in use and $Q = 4$, we cannot observe the same behavior under the same penalty type with $Q = 5$ and $Q = 6$.

We observe that the majority of the contributions made to MAP levels come from the states in which the best action is to transition to the full state in both MDP models. Aggregate MDP model is very good at matching with these type of actions in general. It is not very competitive in anticipating the best action for the states that have several good quality actions available. We believe that part of the reason behind having better APD and MAP values with penalty types 2 and 3 is because they apply penalty levels directly proportional to immediate reward coefficients

$(n_k f_k)$ which in turn restricts the set of rewarding actions for states. This ultimately produces higher chance for the aggregate MDP to identify the best action taken by the MDP and approximate the total expected rewards.

We report problem setup times and value iteration algorithm solution times for both MDP and aggregate MDP models in Table 3.7. We run these test problems on a Core 2 Quad 2.93 GHz, 4 GB RAM computer. We see that solving the MDP model takes significantly more time than solving the aggregate MDP model in all instances. This is a direct result of including larger number of states ($|S|$) in the MDP compared to the one ($|K| + 2$) in the aggregate MDP. However, as the number of states increases, it takes dramatically more computation time to set up the aggregate MDP problem components. We also tested the computational performance of the two models on the problem instances with very large number of states as seen in Table 3.8. The aggregate MDP model is computationally more efficient than the original MDP model although the setup time for the larger problems is high. These experiments are performed on a computer with two Intel six-core Xeon X5670 2.93 GHz processors and 24GB of memory.

Table 3.7: Computational Performance of MDP and Aggregate MDP models

				MDP		Aggregation						MDP		Aggregation	
Penalty Type	Q	$\|K\|$	$\|S\|$	Setup (sec.)	Algorithm (sec.)	Setup (sec.)	Algorithm (sec.)	Penalty Type	Q	$\|K\|$	$\|S\|$	Setup (sec.)	Algorithm (sec.)	Setup (sec.)	Algorithm (sec.)
1	4	5	57	0.026	1.899	0.097	0.282	1	4	8	166	0.175	6.800	0.403	0.482
2	4	5	57	0.004	1.493	0.068	0.200	2	4	8	166	0.105	7.268	0.340	0.311
3	4	5	57	0.003	1.423	0.046	0.143	3	4	8	166	0.041	7.174	0.269	0.330
1	5	5	127	0.008	5.236	0.160	0.212	1	5	8	496	0.175	64.324	1.224	0.616
2	5	5	127	0.012	5.273	0.144	0.182	2	5	8	496	0.181	64.245	1.292	0.621
3	5	5	127	0.006	5.037	0.122	0.219	3	5	8	496	0.181	64.141	1.265	0.615
1	6	5	253	0.023	22.049	0.277	0.288	1	6	8	1288	1.045	580.842	5.519	1.278
2	6	5	253	0.022	22.465	0.314	0.243	2	6	8	1288	1.082	598.189	5.859	1.304
3	6	5	253	0.022	22.593	0.305	0.268	3	6	8	1288	1.071	577.072	5.730	1.341
1	4	6	85	0.048	3.140	0.167	0.247	1	4	9	221	0.271	11.172	0.683	0.475
2	4	6	85	0.006	2.673	0.075	0.158	2	4	9	221	0.190	11.460	0.309	0.324
3	4	6	85	0.004	2.486	0.089	0.163	3	4	9	221	0.115	11.620	0.357	0.377
1	5	6	211	0.021	12.605	0.257	0.249	1	5	9	716	0.630	130.615	2.172	0.858
2	5	6	211	0.019	12.511	0.247	0.303	2	5	9	716	0.610	129.702	2.199	0.842
3	5	6	211	0.019	12.289	0.282	0.256	3	5	9	716	0.596	131.751	2.181	0.813
1	6	6	463	0.091	69.813	0.952	0.455	1	6	9	2003	3.843	1427.650	11.937	2.154
2	6	6	463	0.091	69.963	0.916	0.452	2	6	9	2003	3.758	1444.031	12.299	2.157
3	6	6	463	0.089	69.728	0.955	0.419	3	6	9	2003	3.658	1446.783	11.780	2.141
1	4	7	121	0.100	4.664	0.246	0.220	1	4	10	287	0.884	18.449	1.082	0.498
2	4	7	121	0.140	4.249	0.184	0.250	2	4	10	287	0.386	18.473	1.060	0.423
3	4	7	121	0.109	4.123	0.146	0.234	3	4	10	287	0.415	18.351	1.197	0.465
1	5	7	331	0.060	26.009	0.535	0.362	1	5	10	1002	2.333	251.040	4.091	1.140
2	5	7	331	0.057	25.929	0.568	0.385	2	5	10	1002	2.358	252.854	4.018	1.211
3	5	7	331	0.058	25.789	0.610	0.426	3	5	10	1002	2.355	253.151	3.911	1.229
1	6	7	793	0.321	207.925	2.287	0.736	1	6	10	3004	13.449	3272.205	26.170	3.473
2	6	7	793	0.321	200.939	2.264	0.721	2	6	10	3004	13.191	3273.545	27.323	3.575
3	6	7	793	0.321	200.400	2.321	0.715	3	6	10	3004	15.575	3214.341	33.887	3.608

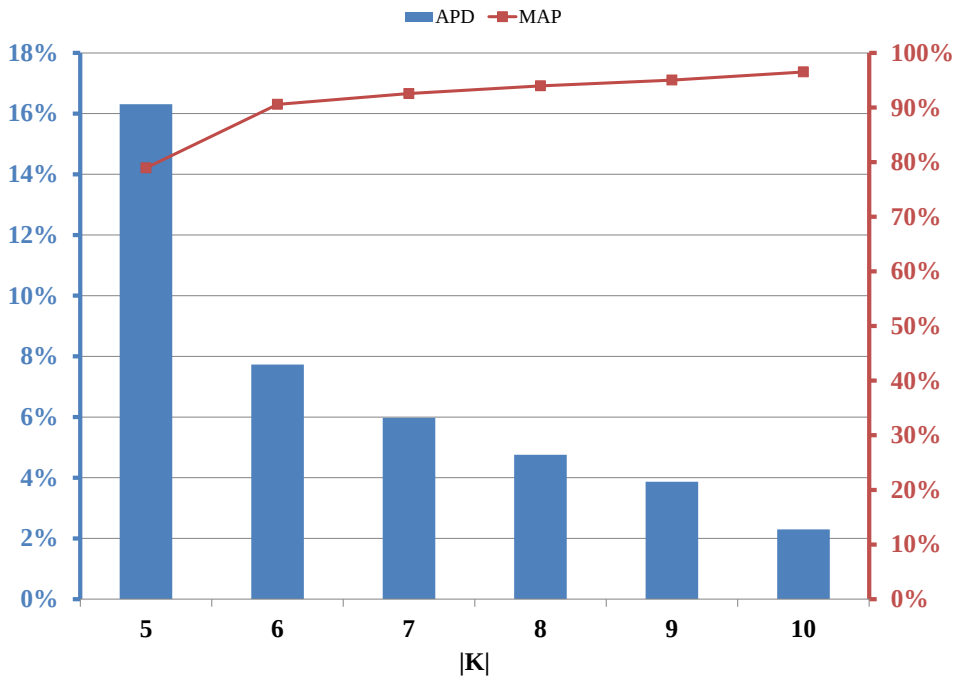

Figure 3.1: APD and MAP with penalty type 1 and $Q = 4$

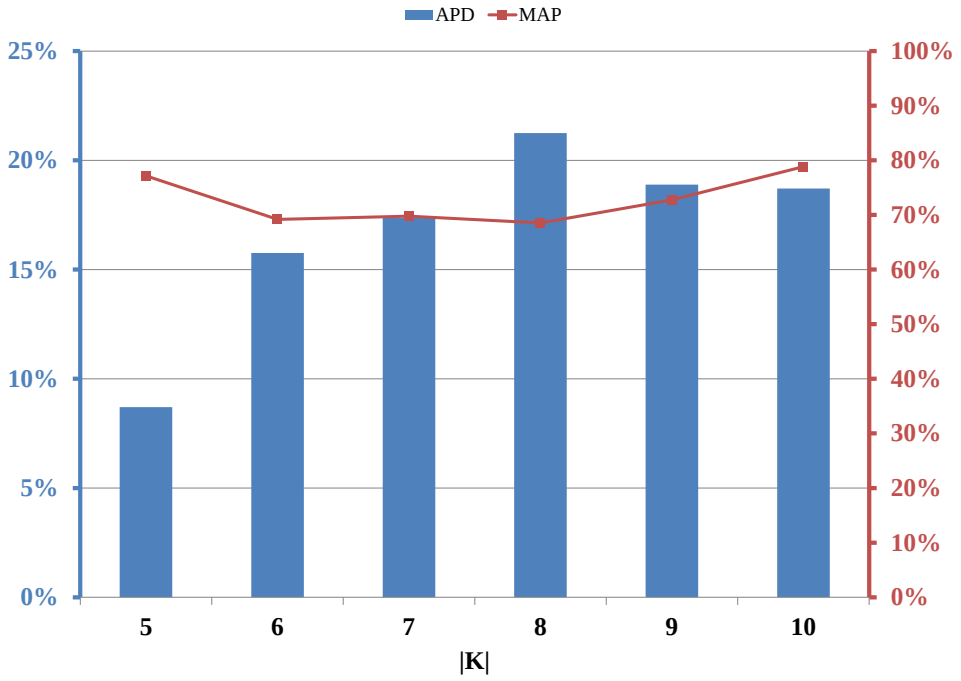

Figure 3.2: APD and MAP with penalty type 1 and $Q = 5$

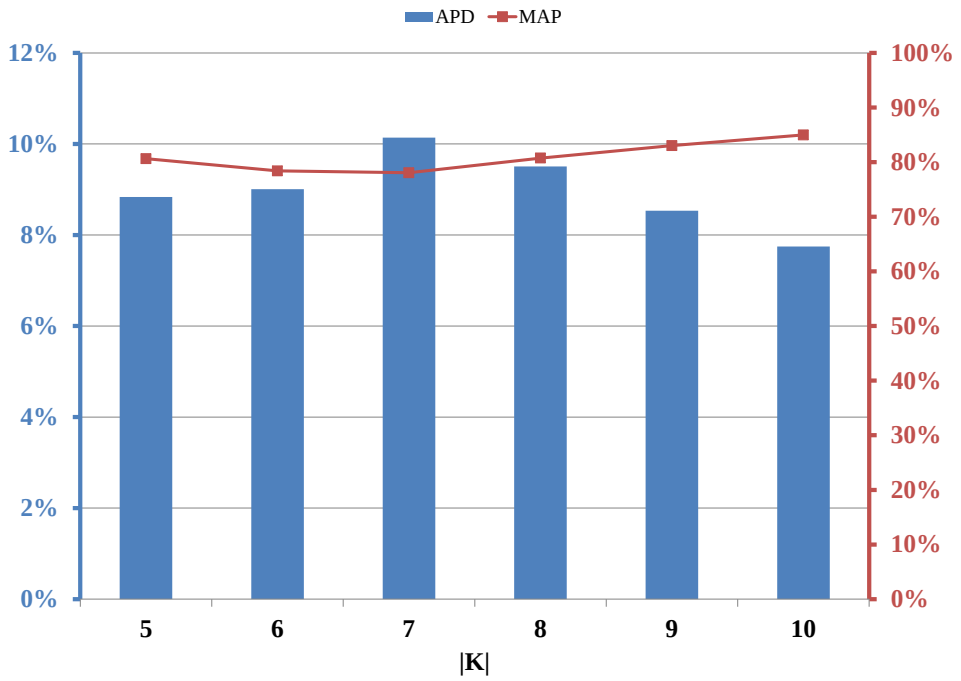

Figure 3.3: APD and MAP with penalty type 1 and $Q = 5$

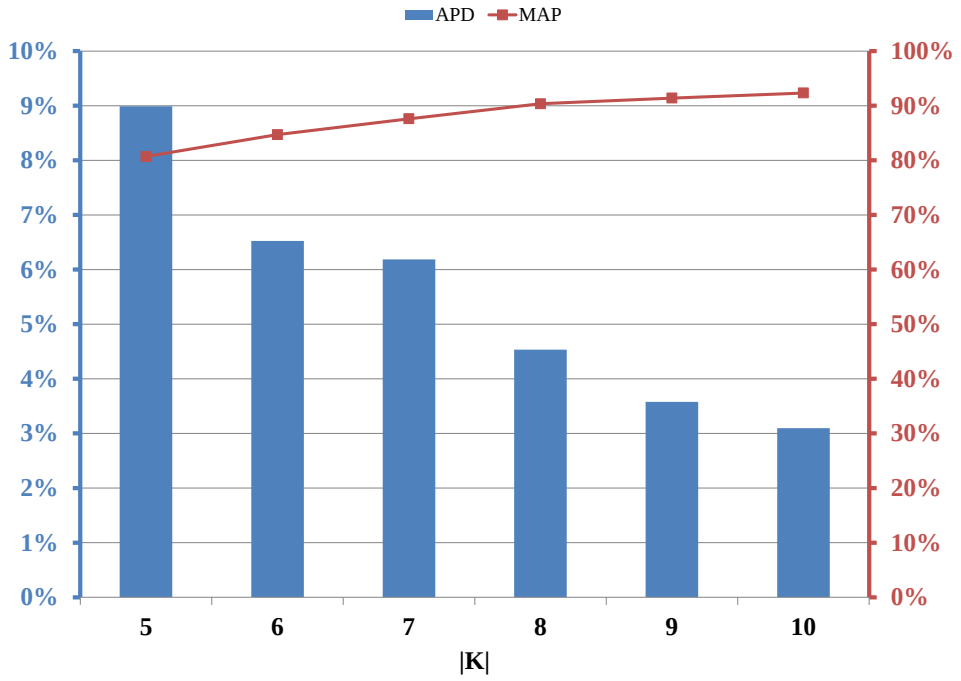

Figure 3.4: APD and MAP with penalty type 2 and $Q = 4$

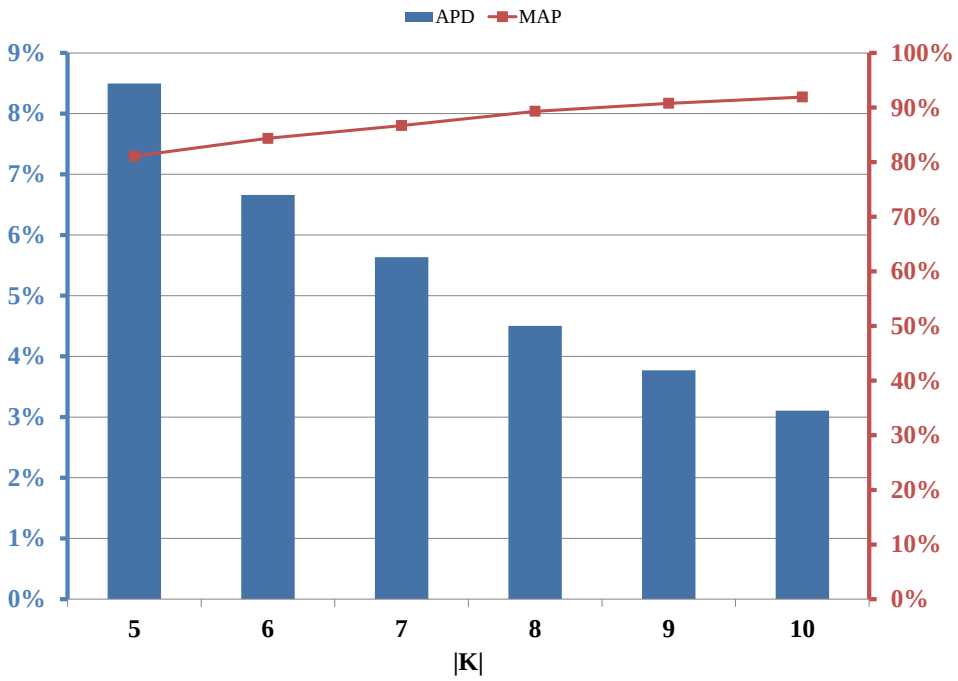

Figure 3.5: APD and MAP with penalty type 2 and $Q = 5$

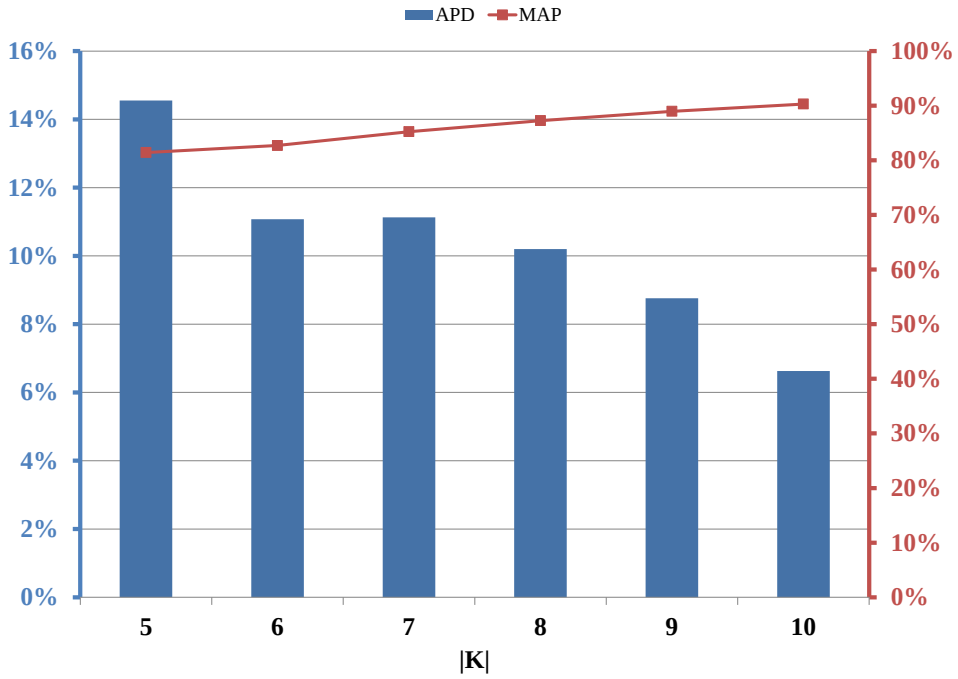

Figure 3.6: APD and MAP with penalty type 2 and $Q = 6$

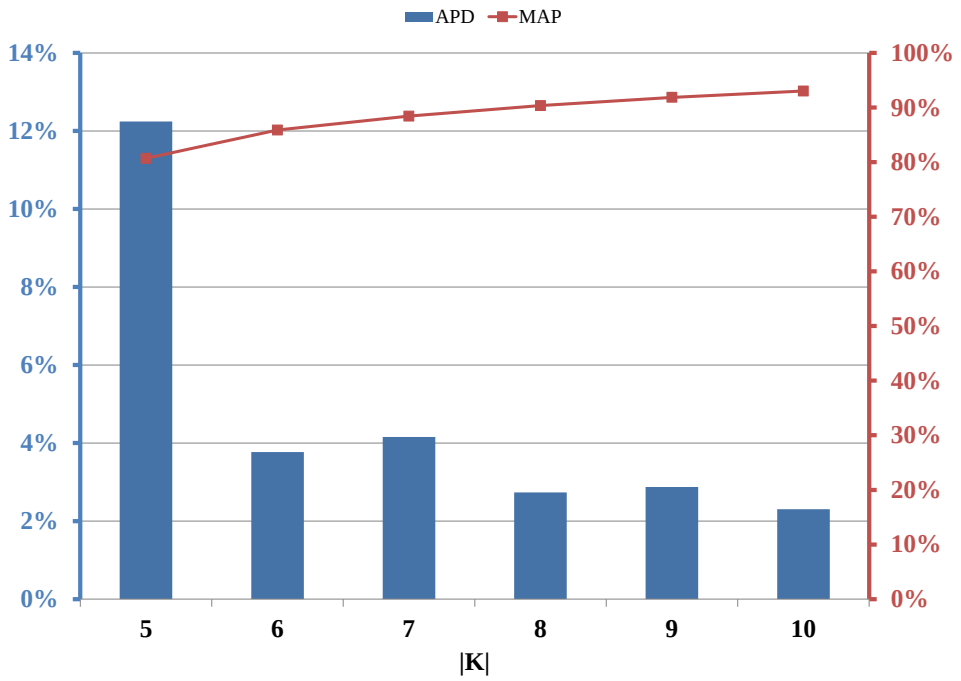

Figure 3.7: APD and MAP with penalty type 3 and $Q = 4$

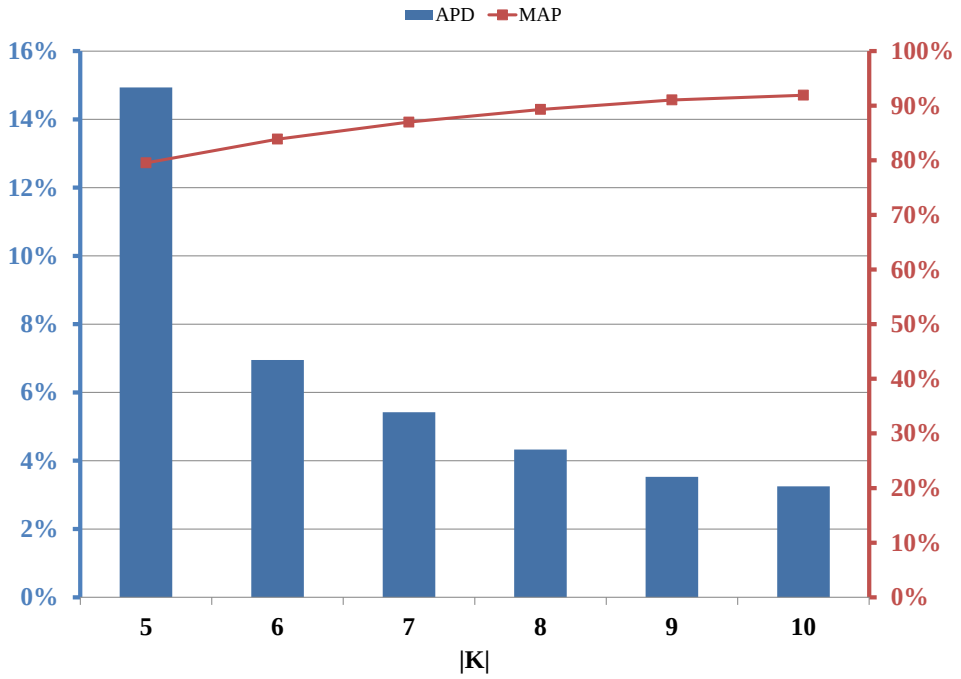

Figure 3.8: APD and MAP with penalty type 3 and $Q = 5$

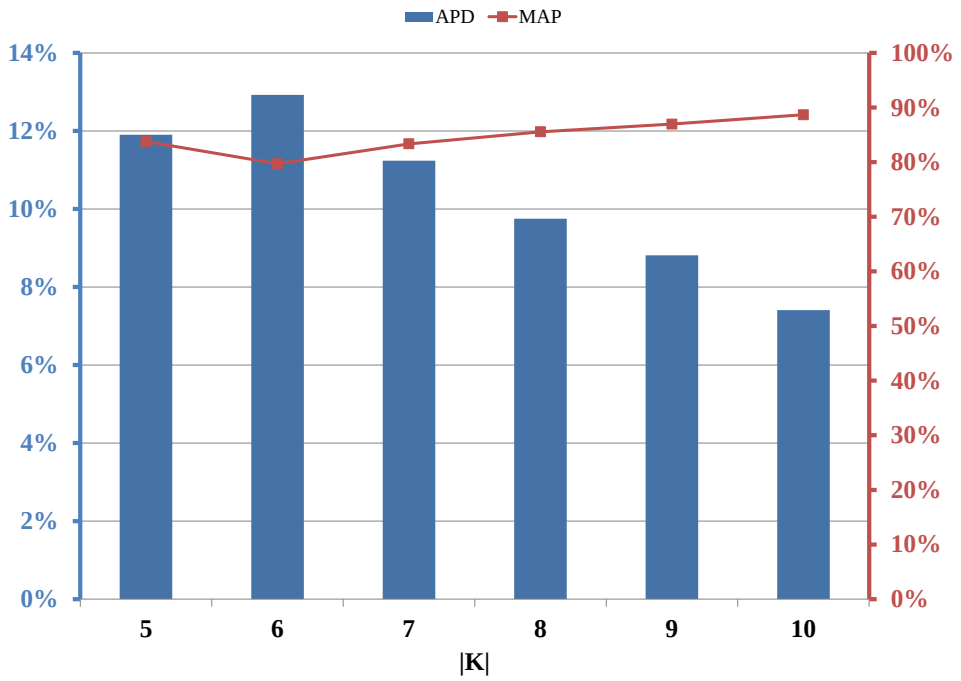

Figure 3.9: APD and MAP with penalty type 3 and $Q = 6$

Table 3.8: Performance of (aggregate) MDP models on large problem instances

			MDP		Aggregation	
Q	$\|K\|$	Number of states	Setup (sec.)	Algorithm (sec.)	Setup (sec.)	Algorithm (sec.)
7	10	8009	56.93	19111.621	128.065	4.709
15	5	11629	-	-	3900.989	7.43
8	10	19449	-	-	865.857	12.243
17	5	20350	-	-	3900.989	7.430
20	5	42505	-	-	30607.492	16.096
10	10	92379	-	-	32868.907	65.671

3.5 Conclusions and Future Research Directions

The MDP model proposed in this chapter is powerful decision mechanisms in the case of an unexpected opening in the facility (i.e. no-shows, appointment cancellations etc.). It aims to find the best patient admission decision with the maximum total expected reward while conforming to the patient mix and capacity restrictions. Numerical experiments demonstrate that it becomes intractable due to exponentially growing state space as problem parameters increase. Thus, approx-

imate optimal patient admission policies are targeted via an aggregate MDP model developed by fixed-weight aggregation technique. Despite its ease of application, it is capable of providing very good executable approximate policies for the original MDP model which is also evidenced by small APD and large MAP values. Note that worst case APD (MAP) values are 21% (69%), 14.1% (79%) and 14.4% (80%) for the penalty type 1,2, and 3, respectively.

There are several future directions regarding this study. Note that gantry capacity constraints are the only factor that limits the acceptance of new patients in the MDP model. One important extension of this work would be incorporating other operational restrictions such as sequencing of patients, anesthesia surveillance team availability and assessing the impacts of these on patient admission policies. For instance, in order to account for the treatments of patients who need other resources (i,e, technicians or nurses) during their treatment sessions, we need to differentiate the time slots based on the availability of such resources in gantries. This would dramatically increase the dimension of both the state and action spaces which in turn ultimately impact the complexity of the problem. Another important extension of this work would be accounting for the waiting patients for this treatment. Since this is another layer of information that describes the state of the facility, it should be incorporated in the state space definition. Finally, a more competitive aggregation technique that can mimic the behaviors of the MDP model with several other operational constraints would be another possible research direction.

Appendix

3.A Work Verification Letter

College of Engineering
Department of Industrial Engineering

Date: July 8, 2014

Graduate School
University of Arkansas

Dear Dr. Needy:

I am writing to verify that Ridvan Gedik completed more than 51% of the work for the chapter titled "Proton Therapy Patient Scheduling: Markov Decision Process Modeling," in his dissertation. He is also the first author of this article.

Sincerely,

Chase Rainwater
cer@uark.edu
479-575-2687
Assistant Professor
Department of Industrial Engineering
University of Arkansas

Bibliography

Adan, I. and Vissers, J. (2002). Patient mix optimisation in hospital admission planning: a case study. *International Journal of Operations & Production Management*, 22(4):445–461.

American Cancer Society (2014). Cancer Facts and Figures 2014. `http://www.cancer.org/acs/groups/content/@research/documents/webcontent/acspc-042151.pdf`. [Online; accessed 30-June-2014].

Andre, D. and Russell, S. J. (2002). State abstraction for programmable reinforcement learning agents. In *AAAI/IAAI*, pages 119–125.

Bertsimas, D. and Demir, R. (2002). An approximate dynamic programming approach to multidimensional knapsack problems. *Management Science*, 48(4):550–565.

Bowers, J. and Mould, G. (2005). Ambulatory care and orthopaedic capacity planning. *Health Care Management Science*, 8:41–47.

Cayirli, T., Veral, E., and Rosen, H. (2006). Designing appointment scheduling systems for ambulatory care services. *Health Care Management Science*, 9:47–58.

Conforti, D., Guerriero, F., and Guido, R. (2008). Optimization models for radiotherapy patient scheduling. *4OR*, 6:263–278.

Conforti, D., Guerriero, F., and Guido, R. (2010). Non-block scheduling with priority for radiotherapy treatments. *European Journal of Operation Research*, 201:289–296.

Dean, T., Givan, R., and Leach, S. (1997). Model reduction techniques for computing approximately optimal solutions for markov decision processes. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pages 124–131. Morgan Kaufmann Publishers Inc.

Dearden, R. and Boutilier, C. (1997). Abstraction and approximate decision-theoretic planning. *Artificial Intelligence*, 89(1):219–283.

Dietterich, T. G. (2000). Hierarchical reinforcement learning with the maxq value function decomposition. *Journal of Artificial Intelligence Research*, 13(1):227–303.

Gocgun, Y., Bresnahan, B. W., Ghate, A., and Gunn, M. L. (2011). A markov decision process approach to multi-category patient scheduling in a diagnostic facility. *Artificial Intelligence in Medicine*, 53(2):73 – 81.

Heyman, D. P. and Sobel, M. J. (2003). *Stochastic Models in Operations Research: Stochastic Optimization*, volume 2. Courier Dover Publications.

Kapadia, A. S., Vineberg, S. E., and Rossi, C. D. (1985). Predicting course of treatment in a rehabilitation hospital: a markovian model. *Computers & OR*, 12(5):459–469.

Kapamara, T., Sheibani, K., Haas, O., Reeves, C., and Petrovic, D. (2006). A review of scheduling problems in radiotherapy. In *Proceedings of the eighteenth international conference on systems engineering (ICSE2006), Burnham KJ, Haas OCL, Coventry University, UK*, pages 201–207.

Kolesar, P. (1970). A markovian model for hospital admission scheduling. *Management Science*, 16(6):pp. B384–B396.

Li, L., Walsh, T. J., and Littman, M. L. (2006). Towards a unified theory of state abstraction for mdps. In *ISAIM*.

Nunes, L. G. N., de Carvalho, S. V., and Rodrigues, R. d. C. M. (2009). Markov decision process applied to the control of hospital elective admissions. *Artificial intelligence in medicine*, 47(2):159–171.

Pham, D. N. and Klinkert, A. (2008). Surgical case scheduling as a generalized job shop scheduling problem. *European Journal of Operation Research*, 185:1011–1025.

Powell, W. B. (2007). *Approximate Dynamic Programming: Solving the curses of dimensionality*. John Wiley & Sons.

Powell, W. B. (2009). What you should know about approximate dynamic programming. *Naval Research Logistics (NRL)*, 56(3):239–249.

Powell, W. B., George, A., Bouzaiene-Ayari, B., and Simao, H. P. (2003). Approximate dynamic programming for high dimensional resource allocation problems. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 5, pages 2989–2994. IEEE.

Powell, W. B., Simao, H. P., and Bouzaiene-Ayari, B. (2012). Approximate dynamic programming in transportation and logistics: a unified framework. *EURO Journal on Transportation and Logistics*, 1(3):237–284.

Sauré, A., Patrick, J., Tyldesley, S., and Puterman, M. L. (2012). Dynamic multi-appointment patient scheduling for radiation therapy. *European Journal of Operational Research*, 223(2):573–584.

Schütz, H.-J. and Kolisch, R. (2012). Approximate dynamic programming for capacity allocation in the service industry. *European Journal of Operational Research*, 218(1):239–250.

Sutton, R. S., Precup, D., and Singh, S. (1999). Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1):181–211.

4. ANALYSIS OF A PARALLEL MACHINE SCHEDULING PROBLEM WITH SEQUENCE DEPENDENT SETUP TIMES AND TIME WINDOWS

Ridvan Gedik Chase Rainwater Heather Nachtmann Edward Pohl

4.1 Introduction

This study aims to provide useful insights for decision-makers managing a fleet of resources, which is composed of a limited number of dissimilar machines under operational and tactical level restrictions. For a given time horizon, we seek to assign nonidentical jobs to machines and provide the best sequence for each piece of machine. During the search for the best assignment and job sequence, one must also account for the time spent navigating in between jobs. Moreover, since the machines are dissimilar, completion time of each job depends on the type of the machine assigned to it. Finally, each job has a cost, profit, time availability window(s) and can be assigned to at most one machine. The total cost of operations across all jobs has to be less than or equal to a budget. Subject to all these requirements, a typical decision-maker's objective is to either (i) maximize the total profit or (ii) minimize the makespan within a given time horizon. The main differences between modeling this problem with two different objectives can be seen in Section 4.2. In order to clarify some other aspects of the problem description, the following assumptions are made in all models given in this study:

- Release time and deadline of each job are start and end of planning horizon, respectively.
- All jobs are available for processing at the beginning of time horizon.
- Tardy jobs are not allowed.
- A piece of machine can work on at most one job at a time.
- A job can be assigned to at most one machine.
- Job preemption is not allowed.
- Job processing times are deterministic, but vary based on the machine type.

- Sequence dependent travel (setup) time from one job location to another is deterministic and assumed to be same for each machine.
- There is no travel (setup) time before the first job and after the last job in a machine's schedule.
- A job might have more than one time availability window restriction. A restricted period is defined as a complementary concept for time availability window to represent the times when job processing is prohibited.

The remainder of the paper is organized as follows. Section 4.2 discusses the integer programming (IP), constraint programming (CP) and decomposition algorithms developed to solve our scheduling problem. A real world application of this problem with corresponding instances is introduced in Section 4.3. Finally, Section 4.4 compares the performance of the proposed solution techniques.

4.2 Problem Modeling: Mathematical versus Constraint Programming

In recent years, constraint programming has been widely applied to a variety of scheduling problems as an alternative or in conjunction with IP. From a general perspective, the first important distinction between constraint programming and mathematical programming (specifically, ILP, integer linear programming) is the ways that they express the conditions to solve an optimization or a constraint satisfaction problem. On one hand, CP allows declarative, flexible and compact model formulations which makes adapting new constraints straightforward while not affecting the previous constraints (Focacci et al., 2002; Hooker, 2007a). Especially, the ability of representing complex application oriented constraints in terms of *global (logical) constraints* brings CP a competitive advantage when the actual interest is to provide good feasible solutions (Jain and Grossmann, 2001). Hence, CP has proven to have good performance in solving highly constrained discrete optimization and feasibility problems such as scheduling, planning and resource allocation problems. On the other hand, ILP is more likely to perform better when the attacked

problem has a pure (well studied) geometrical structure compared to the CP (Focacci et al., 2002) and (Lombardi and Milano, 2012). It should be noted that when the side constraints are added, the problems become less pure, and the application of CP seems to be more appropriate.

A recent increase in the number of studies which use CP to develop (part of) solution techniques for classical operations research problems suggest a growing interest in its functionality. Jain and Grossmann (2001) compare pure CP and ILP model formulations with a hybrid CP/ILP model for a scheduling problem which involves dissimilar parallel machines, nonidentical orders with different release times and deadlines. They demonstrated that the hybrid model is able to solve larger instance as opposed to pure CP and ILP models. Besides, the decomposition (Logic-Based Benders) algorithm outperforms all other alternatives in terms of solution time and objective function value. Later, Sadykov and Wolsey (2006) studied the same problem in order to generate efficient decomposition algorithms based on column generation (branch and price) and branch and cut approaches. Harjunkoski and Grossmann (2002) also studied the parallel scheduling problem motivated by Jain and Grossmann (2001) with sequence independent setup times and developed another decomposition method in which both master and subproblems are formulated as ILP. More details on CP/ILP based decomposition algorithms and hybrid modeling approach are included in Section 4.2.3.

For some other versions of parallel machine scheduling problems, a double or combined modeling approach is reported to be an efficient solution method. Edis and Ozkarahan (2011) developed a combined ILP/CP model formulation which tackles "resource-constrained identical parallel machine scheduling problem with machine eligibility restrictions" and showed that the combined ILP/CP model is able to find the optimal solution in 174 out of 200 test instances with substantial decreases in solution time as opposed to solving the same test instances by pure ILP and CP models. Moreover, pure ILP and CP models managed to find optimal solutions in only 47 and 6 problems, respectively. Similar success of combined ILP/CP approach is reported in Edis and Oguz (2012) which tackles parallel machine scheduling with flexible resources problem. There are other operations research problems that are successfully solved by combined ILP/CP

approach compared to pure ILP and CP methods such as time-tabling/rostering (Trick and Yildiz, 2011; Topaloglu and Ozkarahan, 2011; He and Qu, 2012), project scheduling with time windows (Cesta et al., 2002). For further information about double or hybrid modeling and other decomposition methods, we refer the reader to Hooker (2006b) and van Hoeve and Katriel (2006).

Pearn et al. (2002) studied a parallel machine scheduling problem with nonidentical job processing times, identical machines and sequence dependent setup times in a wafer probing factory. They proposed an ILP model and then its transformation to a well studied vehicle routing problem with time windows due to the complexity of the ILP model formulation. Three different heuristics were used to find the near-optimal solutions for a real-world test instance with 100 jobs, 14 identical equipments and 30 different product types. Cakici and Mason (2007) also considered parallel scheduling problem in semiconductor manufacturing with auxiliary resource constraints and proposed a heuristic which produces a solution with a 0.78% optimality gap. A comprehensive survey on scheduling problems in semiconductor manufacturing operations is proposed in Mönch et al. (2011).

Rojanasoonthon and Bard (2005) addressed a parallel machine scheduling problem with time windows, priority levels on jobs and with the objective of maximizing the number of jobs scheduled. They reported that their ILP is unable to report even a feasible problem and therefore, they provided a greedy randomized adaptive search procedure that produced good quality solutions for the data instances with 400 jobs and 6 machines. A different variant of parallel scheduling problem was first described by Arkin and Silverberg (1987) in which each job has a fixed start time, end time and a value. The objective is to maximize the total value associated with the feasible subset of jobs that are to be processed. This problem with dissimilar equipments is shown to be NP-complete. Türsel Eliiyi and Azizoğlu (2009) addressed the fixed job scheduling problem with machine dependent job values and developed a branch and bound algorithm that is capable of obtaining optimal solutions for the large instances with 100 jobs. We refer the reader to Allahverdi et al. (2008) which provided a comprehensive review on scheduling problems with numerous constraints and performance measures based on due date (makespan, tardiness etc.),

setup time, (weighted) completion or delivery time of jobs and flowtime.

In the light of strengths and weaknesses of previous scheduling models developed in different framework, Sections 4.2.1, 4.2.2 and 4.2.3 elaborate on our modeling efforts on problem described in Section 4.1.

4.2.1 Integer Programming (IP) Formulation

Before explaining the details of the IP, required notation to account for the key components of the scheduling problem is given in Table 4.1.

Table 4.1: Notation

	Sets
$d \in D$	set of machines
$t \in T$	set of consecutive time periods comprising the planning horizon
$j \in J$	set of jobs
$w \in W_j$	set of restricted periods applicable to job j
	Parameters
b_w	the beginning of restricted period $w \in W_j$; $j \in J$
e_w	the end of restricted period $w \in W_j$; $j \in J$
r_d	the operation rate of machine $d \in D$
q_j	the profit associated with job $j \in J$
$t_{jd} = \left\lceil \frac{q_j}{r_d} \right\rceil$	the time it takes for machine $d \in D$ to complete job $j \in J$
$t_{jj'}$	the time that it takes to move machine $d \in D$ from job site $j \in J$ to job site $j' \in J$ ($j \neq j'$)
c_j	the cost for completing job $j \in J$
B	the available budget for the planning horizon
	Decision variables
y_{dj}	1 if machine d is used to complete job j
z_{djt}	1 if machine d begins working on job j in period t
v	makespan

Then, we introduce the mixed integer programming model (DS) as follows.

$$\text{maximize} \sum_{j \in J} \sum_{d \in D} q_j y_{dj}$$

subject to (DS)

$$\sum_{d\in D} y_{dj} \leq 1 \qquad j \in J \tag{4.1}$$

$$\sum_{j\in J}\sum_{d\in D} c_j y_{dj} \leq B \tag{4.2}$$

$$\sum_{t\in T} z_{djt} = y_{dj} \qquad j \in J; d \in D \tag{4.3}$$

$$\sum_{t'=t}^{\min\{|T|,t+t_{jd}+t_{jj'}\}} z_{dj't'} \leq 1 - z_{djt} \qquad j \in J; j' \in J; j \neq j'; d \in D; t \in T \tag{4.4}$$

$$\sum_{d\in D}\sum_{t=\max\{1,b_w-t_{jd}\}}^{e_w} z_{djt} = 0 \qquad w \in W_j; j \in J \tag{4.5}$$

$$\left(t + t_{jd}\right) z_{djt} \leq |T| \qquad j \in J; d \in D; t \in T \tag{4.6}$$

$$y_{dj} \geq 0 \qquad d \in D; j \in J \tag{4.7}$$

$$z_{djt} \in \{0,1\} \qquad d \in D; j \in J; t \in T \tag{4.8}$$

The objective of the model is to maximize the total profit. Constraints (4.1) ensure that job j is satisfied by at most one machine d, whereas constraint (4.2) states that the total cost incurred by such assignment can not exceed the total budget. Constraints (4.3) require that if job j is satisfied by machine d, exactly one start day for that work must be specified for that assignment. Constraints (4.4) specify that if job j is started in period t, by machine d, then machine d cannot begin another job, j', until $t_{jj'} + t_{jd}$ periods have passed (i.e. the time to complete job j on machine d plus the time to travel to job j' from job j). Constraints (4.5) prevent a job from beginning, or ending, on a day that overlaps with a restricted period. Constraints (4.6) ensure that if a job is decided to be processed, the completion time should be before the end of the planning horizon. Finally, constraints (4.7)-(4.8) specify the appropriate domain of each variable in the model.

The same problem where minimizing the makespan is the objective is also formulated in (DS-M). In order to represent the fact that each job has to be processed by a single machine, the assignment constraint is modified as in (4.9). Since the budget is not a concern anymore, con-

straint (4.2) is not included in (DS-M). Finally, constraints (4.10) are used to enforce that the makespan is greater than or equal to the maximum of job completion time.

$$\text{minimize} \quad v$$

subject to (DS-M)

$$\sum_{d \in D} y_{dj} = 1 \qquad j \in J \tag{4.9}$$

$$v \geq \left(t + t_{jd}\right) z_{djt} \qquad j \in J; d \in D; t \in T \tag{4.10}$$

$$(4.3), (4.4), (4.5), (4.7), (4.8)$$

$$v \geq 0 \tag{4.11}$$

As with many integer programs, providing the exact optimal schedules for each machine and for each job gets more challenging as the number of decision variables and constraints increase. Given the difficulty in solving problems (DS) and (DS-M) in an IP context and the strengths of CP in finding feasible solutions for highly constrained problems by *global constraints* and *interval variables*, the next sections discuss the CP model formulation equivalent to (DS) and (DS-M) and then discuss decomposition algorithms which enhance the competitive advantages of both CP and IP.

4.2.2 Constraint Programming Approach

4.2.2.1 Search in Constraint Programming

The effectiveness of CP can be assessed by focusing on the constraint and variable definition choices a modeler explores. Heipcke (1999) pointed out that CP models, in general, include much more specific information about variables, constraints and the relationships between/among them. Primarily this factor allows developing stronger and more efficient specialized solution strategies for highly constrained complex problems. Conveying information between constraints

and variables is made possible by *constraint propagation (filtering)* iterative processes of global constraints. Each global constraint is associated with a propagation algorithm to remove the values of variables from their domains that prevent constraints from being feasible when they are assigned to a variable (van Hoeve and Katriel, 2006; Hooker, 2006b). The propagation algorithm of a constraint is run each time a change occurs on a variable (for instance a value is removed form the domain). Since constraints are related to each other through shared variables, whenever a change occurs on the domain of a shared variable due to the propagation algorithm of a constraint, the filtering algorithms of other constraints are also triggered to evaluate possible other reductions in the domains of all variables (Lombardi and Milano, 2012; Harjunkoski and Grossmann, 2002; van Hoeve and Katriel, 2006). Once all possible reductions on domains are made and a feasible solution has not been found, branching on a variable takes place. At this point, one can see that addition of new constraints at any moment does not impact the current model or the search since the propagation algorithms of the previous constraints will remain unchanged due to *incremental search* (Focacci et al., 2002).

For further information on details of the search (branching rules, backtracking, dead end etc.) in constraint programming, we refer the reader to Hooker (2002), Heipcke (1999), van Hoeve and Katriel (2006) and Hooker (2006b) which provide rich descriptions and clear differences of search trees in details used in both ILP and CP.

4.2.2.2 Constraint Programming Models

In addition to the notation in Table 4.1, the following parameters and decision variables are used in developing the CP formulation as well.

Parameters

- $I(j)$ is the *step function* of job $j \in J$. That is $I(j) = 0\%$, if the job j is not allowed to be processed at time t such that $b_w \leq t \leq e_w$, $I(j) = 100\%$ otherwise.

- $TD(t_{jj'})$ is the *transition distance function* between job $j \in J$ and $j' \in J$. It is used to inform

other global constraints that the travel time between job pairs j and j' should be at least $t_{jj'}$.

Decision variables

- Y_{jd}, optional interval variable when job $j \in J$ is assigned to machine $d \in D$ with job duration of t_{jd};
- $Y_j = \{Y_{j1}, Y_{j2}, \ldots, Y_{jD}\}$, set of interval variables representing possible machine $d \in D$ that can be assigned to job $j \in J$;
- $Y_d = \{Y_{1d}, Y_{2d}, \ldots, Y_{Jd}\}$, set of interval variables representing possible jobs $j \in J$ that can be assigned to machine $d \in D$ (*interval sequence variable* for d);
- Z_j, optional interval variable associated with job $j \in J$.

An *interval variable* (IBM (2011)) is a powerful way of representing generic decision variables of a scheduling problem. It addresses the time interval of a job that is being processed by explicitly assigning start and end times. One of its important features is that these variables can be *optional* which enables modeling different assignment alternatives in combinatorial problems. For instance, if an interval variable is optional and *absent*, it is not considered in the solution schedule and its domain is left empty. Otherwise, if an optional interval variable is *present*, it implies that it is considered in the solution and its domain should be filtered to a single value represented by a start and end time. The status or Boolean value of an interval variable can be retrieved by using the *presenceOf(Interval Variable)* constraint.

In light of the basic definition of an interval variable, the constraint programming formulation of the parallel machine scheduling problem with maximizing total profit (CP-DS) is given below.

$$\text{maximize} \quad \sum_{j \in J} q_j \, presenceOf(Z_j)$$

subject to (CP-DS)

$$Alternative(Z_j, Y_j) \qquad j \in J \tag{4.12}$$

$$Cumulative(Z_j, c_j, B) \tag{4.13}$$

$$Cumulative(Z_j, 1, |D|) \tag{4.14}$$

$$Z_j.StartMin = 1 \qquad j \in J \tag{4.15}$$

$$Z_j.EndMax = |T| \qquad j \in J \tag{4.16}$$

$$ForbidExtent(Z_j, I(j)) \qquad j \in J \tag{4.17}$$

$$NoOverlap(Y_d, TD(t_{jj'})) \qquad d \in D \tag{4.18}$$

The objective function of (CP-DS) seeks to maximize the total profit in a planning horizon. Constraints (4.12) ensure that each job can only be assigned to at most one machine. *Alternative* constraint enforces that if Z_j is present in the solution, then one and only one of the Y_j will be present in the solution in order to make the assignment decision. Constraint (4.13) assures that the total cost of operations cannot exceed the budget. *Cumulative* constraint is used to model the resource usage over time and computed with the help of its elementary sub-functions such as *Step*, *Pulse*, *StepAtStart* and *StepAtEnd* (IBM (2011)). $StepAtStart(Z_j)$ is used to increase the total money spent on operations at the start of interval variable Z_j by c_j amount. *Cumulative* in constraint (4.13) is utilized for restricting total spending not to exceed the budget at any time. Similarly, the *Cumulative* constraint and $Pulse(Z_j)$ function are used to make sure that total number of occupied machines at any time can not exceed the fleet size ($|D|$) as in constraint (4.14) where $Pulse(Z_j)$ increases and decreases the cumulative usage of fleet by one at the start and end of interval variable Z_j, respectively.

Constraints (4.15) and (4.16) set the minimum start time and maximum end time of each job to the first and last day of the planning horizon, respectively. *ForbidExtent* constraint (4.17) states that if interval variable Z_j is present in the solution, it cannot overlap with the time intervals where its step function is 0%.

NoOverlap constraints (4.18) ensures that the interval sequence variable Y_d which consists of optional interval variables constitutes the order of the non-overlapping intervals for each machine $d \in D$. Moreover, it also has *TransitionDistance* function ($TD(t_{jj'})$) which puts a minimal time ($t_{jj'}$) to be maintained between the end of interval variable Y_{jd} and the start of interval variable $Y_{j'd}$.

The following formulation (CP-DS-M) is the constraint programming formulation of the parallel machine scheduling problem with a minimizing makespan objective. (CP-DS-M) is equivalent to (DS-M).

$$\text{minimize} \quad v$$

$$\text{subject to} \qquad \text{(CP-DS-M)}$$

$$v \geq EndOf(Z_j) \qquad j \in J \qquad (4.19)$$

$$(4.12), (4.14), (4.15), (4.16), (4.17), (4.18)$$

The objective function of (CP-DS-M) seeks to minimize the maximum makespan represented in constraints (4.19). Note that the budget constraint (4.13) is removed from the formulation. In order to assure that each job is processed by a machine, we change Z_j interval variables from *optional* to *compulsory*. Therefore, constraints (4.12) ensure that only one member of the Y_j will be present in the solution since Z_j must be in the solution.

4.2.3 Hybrid Modeling and Decomposition

A powerful aspect of ILP techniques is that the impacts of all constraints are evaluated simultaneously, and therefore it has a *global perspective* while the search tree is being explored (Rodosek et al., 1999; Harjunkoski and Grossmann, 2002). On the other hand, CP propagation algorithms explore the impacts of constraints sequentially through domain reduction of variables (*local perspective*) (Jain and Grossmann, 2001). These two important features, *global vs. local perspectives*, originate from the unique differences in defining models (constraints and variables) with these two techniques. These differences have significant impact on the subsequent search

procedures. The most important bottleneck that arises in the branch-and-bound search tree of the ILP is when handling the integrality constraints. This ultimately might result in evaluating an exponential number of combinations due to the number of subproblems. Furthermore, if the initial gap between the objective value for the optimal solution and the initial relaxed linear subproblem is large, the effectiveness of ILP tends to degrade. In addition to the number of the constraints and variables, representation of complex relationships between/among variables and constraints can be a major difficulty in providing a concise model. This is because ILP can handle only inequality and equality constraints which might be insufficient to represent real life constraints. On the other hand, such a strong restriction on constraint expression is minimized in CP since application-based global constraints can be utilized which express complex relationships in shorter ways. However, one needs to take care in selecting of global constraints since the quality and speed of the domain reduction process at each node depends on the filtering algorithm running behind these global constraints. Since these propagation algorithms are called multiple times during a search, inefficient algorithms might dramatically slow down the search process. It should be remembered that not all global constraints have efficient constraint propagation engines (Jain and Grossmann, 2001). Therefore, hybrid approaches aim to develop integrated methods to merge the complementary strengths of ILP and CP to solve problems that are intractable using either of these two methods alone. Modeling an entire problem in both the CP and ILP contexts is referred to as *double modeling*. Milano (2004); Hooker (2002) and Focacci et al. (2002) explain the advantages and disadvantages of this formulation technique in detail.

More recently, Benders decomposition and Branch & Price algorithms have shown to be very effective when reformulated in a hybrid CP and ILP framework. In the Branch & Price algorithm, the master problem and subproblem are formulated as ILP and CP respectively (Topaloglu and Ozkarahan, 2011; He and Qu, 2012). Using CP as a column generator takes advantage of CP's flexibility to formulate complex relationships that might occur in pricing problems (Hooker, 2006a). Similarly, the classic Benders decomposition master problem is formulated as an ILP and resolved with the Benders cuts generated from the CP formulated subproblems (Hooker, 2006a,

2007b; Jain and Grossmann, 2001).

4.2.3.1 Benders Decomposition: IP/CP Integration

We observe that (CP-DS-M) produces good solutions within a reasonable computational time as shown in Section 4.4. Although (CP-DS) provides feasible solutions for all instances, it fails to report the optimal solution within a specified time limit. In order to overcome this weakness, we propose two novel logic-based Benders decomposition algorithms based on problem formulations (DS) and (CP-DS). Section 4.4 contains results that document the differences between all approaches.

Hooker (2007b) states that classical Benders decomposition (Benders, 1962) is not appropriate for highly combinatorial problems such as scheduling, because it enforces subproblems to be continuous linear or nonlinear programming problems. Therefore, recent studies have focused on implementing logic-based Benders decomposition in which subproblems are discrete feasibility problems and solved to generate Benders cuts. All generated Benders cuts are added to the master ILP problem. After the master problem is solved to optimality, all subproblems are solved and produced cuts are placed in the cut set. If the master problem is solved to optimality and all subproblems are feasible, the solution is optimal to the global problem. However, if there is at least one infeasible subproblem, corresponding cuts are added to the cut set, and the master problem is called to perform the next iteration of the logic-based Benders algorithm.

Given the competitive advantage of ILP in proving optimality through linear relaxation, the objective function of the problem is modeled in the master problem of logic-based Benders decomposition (M1-DS) which contains assignment (4.20), budget (4.21) constraints and Benders cuts (4.22).

$$\text{maximize} \sum_{j \in J} q_j \left(\sum_{d \in D} y_{dj} \right)$$

subject to (M1-DS)

$$\sum_{d\in D} y_{dj} \leq 1 \qquad j \in J \tag{4.20}$$

$$\sum_{j\in J} c_j \left(\sum_{d\in D} y_{dj} \right) \leq B \tag{4.21}$$

$$\sum_{j\in H_d^k} y_{d'j} \leq |H_d^k| - 1 \qquad d' \in \bar{D}_d; k = \{1,2,\ldots,K-1\} \tag{4.22}$$

$$y_{dj} \in \{0,1\} \qquad d \in D;\ j \in J$$

Note that (M1-DS) has no impact on the scheduling and sequencing decisions of jobs with respect to the machine they are assigned to. Therefore, we need to solve $|D|$ independent subproblems to check if the assignments made by (M1-DS) are feasible or not. Hence, let $H_d^k = \{j | y_{dj}^k = 1\}$ be the set of jobs $j \in J$ that are assigned to machine $d \in D$ at k^{th} iteration and define A_{jd} and $\bar{A}_d$ as the **compulsory (not optional)** interval variable associated with job j and machine d and **interval sequence variable** for machine $d \in D$ respectively. Then, the following *feasibility problem* (without an objective function) is formulated in the CP context and solved for each machine d in order to make specific scheduling and sequencing decisions with respect to restricted periods and travel times between jobs.

(S1-DS: Subproblem for each $d \in D$)

$$NoOverlap(\bar{A}_d, t_{jj'})$$

$$A_{jd}.StartMin = 1 \qquad j \in H_d^k$$

$$A_{jd}.EndMax = |T| \qquad j \in H_d^k$$

$$ForbidExtent(A_{jd}, I(j)) \qquad j \in H_d^k$$

$$Cumulative(A_{jd}, 1, 1)$$

Note that if (S1-DS) is feasible for each machine, the solution for both master and subproblems will be the optimal solution to the overall problem. Otherwise, a "Benders cut " will be gen-

erated for each machine where the jobs assigned to it could not be scheduled successfully. Such cuts are first offered by Hooker et al. (1999) and referred as "no good" and later used by Jain and Grossmann (2001) on a parallel machine scheduling problem. These studies report a significant decrease in solution times when logic based Benders decomposition is applied with "no good" when compared to pure CP or ILP formulations. However, "no good" is demonstrated to be not strong enough to handle scheduling problems.

Assume that jobs in H_d^k are not successfully completed by machine d. Also let $\bar{D}_d = \{d' | r_{d'} \leq r_d,\ d, d' \in D, d \neq d'\}$ be the set of machine operation rates which are less than or equal to the operation rate of $d \in D$. Then, following cuts are formed;

$$\sum_{j \in H_d^k} y_{dj} \leq |H_d^k| - 1 \tag{4.23}$$

$$\sum_{j \in H_d^k} y_{d'j} \leq |H_d^k| - 1 \qquad d' \in \bar{D}_d. \tag{4.24}$$

Note that "no good" (4.23) can be satisfied by omitting just one job from the set H_d^k in later iterations. Moreover, constraints (4.23) cannot prevent assigning same jobs in H_d^k to another machine $d' \in \bar{D}_d$. In order to prevent this, we developed the cuts in (4.24). Thus, if the subproblem for machine d at iteration k is infeasible, then inequalities (4.24) are added to (M1-DS) to enforce that all jobs in H_d^k cannot be assigned to machines $d' \in \bar{D}_d$ for the later iterations. Note that such inequalities cut off several assignment combinations which might only be revealed by several branching/propagation/instantiation operations in a CP search when the whole problem is approached by a CP formulation. Moreover, since subproblems are solved independently in this decomposition approach, it is easy to identify which subproblem, therefore machine, produces infeasible solutions under which assignment scheme. The size of the master problem increases as the the algorithm generates cuts. However, the size of each subproblem cannot exceed the total number of jobs, $|J|$. As a consequence, if the optimal solution is not obtained in early iterations of the algorithm, the time required for solving (M1-DS) to optimality will increase significantly. In order to lessen the intensity of the Benders cut added to (M1-DS), we developed a second

logic-based Benders decomposition algorithm. The master and subproblem of this alternative approach are given below as (M2-DS) and (S2-DS) respectively.

$$\text{maximize} \sum_{j \in J} q_j x_j$$

subject to (M2-DS)

$$\sum_{j \in J} c_j x_j \leq B \tag{4.25}$$

$$\sum_{j \in G^k} x_j \leq |G^k| - 1 \qquad k = \{1, 2, \ldots, K-1\} \tag{4.26}$$

$$x_j \in \{0,1\} \qquad j \in J$$

where x_j is 1 if job j is completed by any machine, 0 otherwise and $G^k = \{j | x_j^k = 1\}$ is the set of jobs that are decided to be processed at iteration k. Similar to (M1-DS), the objective function is to maximize the total profit subject to budget constraint (4.25) and Benders cuts (4.26). Note that (M2-DS) is a *binary knapsack problem* without the Benders cuts. Since assignment decisions are not made by (M2-DS), they have to be handled in (S2-DS) through *optional interval variables*. Hence, let Y_{jd} be the optional interval variable when job $j \in G^k$ is assigned to machine $d \in D$ at iteration k and Y_d be the interval sequence variable for machine $d \in D$. Therefore, we can write the subproblem as

(S2-DS)

$$Alternaive(Z_j, Y_j) \qquad j \in G^k \tag{4.27}$$

$$Z_j.StartMin = 1 \qquad j \in G^k \tag{4.28}$$

$$Z_j.EndMax = |T| \qquad j \in G^k \tag{4.29}$$

$$ForbidExtent(Z_j, I(j)) \qquad j \in G^k \tag{4.30}$$

$$NoOverlap(Y_d, t_{jj'}) \qquad d \in D \tag{4.31}$$

$$Cumulative(Z_j, 1, |D|) \qquad j \in G^k \tag{4.32}$$

where Z_j is the compulsory interval variable for job $j \in G^k$ and $Y_j = \{Y_{j1}, Y_{j2}, \ldots, Y_{jD}\}$ is the set of interval variables Y_{jd} for each $j \in G^k$. Similar to problem (CP-DS), (S2-DS) controls the job assignments to machines. However, in this case, Z_j is a compulsory interval variable that must be present in the solution because M2-DS determines the set of jobs that must be processed. Therefore, constraints (4.27) assure that each job $j \in G^k$ is processed by exactly one machine. Constraints (4.28) and (4.29) enforce that each job must be started and finished within the given planning horizon. Moreover, constraints (4.30) forbid the restricted periods of each job $j \in G^k$ represented by intensity function ($I(j)$) overlapping with its processing time. Similarly, constraints (4.31) make sure that machine d cannot operate while traveling between jobs j and j' such that $j, j' \in G^k$ and $j \neq j'$. Finally, (4.32) is an additional constraint to strengthen the formulation by ensuring that the total number of machines in operation cannot exceed the fleet size at any given time. Note that there are only two problems that need to be solved at each iteration in the second Benders decomposition algorithm. However, the size of subproblem (S2-DS) is significantly larger than the ones represented by (S1-DS). Performance of these two algorithms are evaluated in Section 4.4.

4.3 Case Study: Optimizing Inland Waterway Infrastructure Maintenance for Supply Chain Operations

Each year the U.S. Army Corps of Engineers (USACE) dredges hundreds of navigation projects through its fleet of government dredges and individual contracts with private industry. The decision of assigning dredge resources (government and private industry) to navigation projects is predominately made regionally by awarding the contract to the lowest cost bid that meets the scheduling demands of the dredge job. Most likely, efficiencies can be gained by optimizing

the entire portfolio of dredging jobs. The proposed models and solution approaches in Section 4.2 are used to optimize the decision of allocating dredge resources to projects under necessary constraints such as environmental windows, dredge resource cost and availability, and sequence dependent travel times. Using these approaches, sensitivity analysis on resource levels are performed in order to demonstrate under which circumstances USACE can complete the entire dredging portfolio while achieving compliance and desired system performance.

A specific challenge investigated in this study is the concept of environmental window restrictions. The USACE describes environmental windows as temporal constraints placed upon dredged material disposal operations in order to protect biological resources or their habitats from potentially detrimental effects (Dickerson et al. (I998)). The USACE has documented an increase in total dredging cost without a proportionate increase in total volume of material dredged (Pointon, 1996). Dickerson et al. (I998) stated that a widely-held explanation for this increase in dredging costs is system inefficiencies associated with environmental window compliance. In order to be compatible with the related model constraints, we define restricted periods for each environment window.

Dredge fleet scheduling and sequencing optimization is challenging due to the highly variable and uncertain feature of natural processes, engineering capacity, dredging operations and economic conditions (Ratick and Garriga, 1996). Despite its difficulty level, risk and reliability based dredging optimization papers (see Ratick and Garriga (1996); Menon and Lansey (1990); Lund (1990)) are frequently seen in the literature that model the volatile river and environmental situations that influence dredging operations at a specific project or reach level. The solution techniques in this study provide a system wide optimization perspective which can efficiently and effectively use resources across the entire dredging project portfolio.

Historical USACE dredge project data collected between 1997 and 2011 was utilized to parameterize the model. The data was provided by the Corps Dredging Information System, and a total of 116 unique channel maintenance dredging jobs were identified as seen in Figure 4.1. Table 4.1 demonstrates the descriptive statistics of the model input parameters such as volume of

jobs (q_j), cost of jobs (c_j), length of restricted periods ($e_w - b_w$) and production rate of dredge vessels (r_d). Since the USACE cannot currently afford to meet all the dredging requests, we set the available budget (B) to 75% of the total cost of the 116 jobs. A from-to distance matrix was constructed by using a GIS layer to compute travel distance on the waterways between prospective dredge project locations. Then, the travel (setup) time between each job pair is calculated by assuming the velocity of a dredge vessel is 50 miles per day.

Table 4.1: Descriptive Statistics of Input Parameters

	Average	Minimum	Maximum	St. Dev.
q_j (cubic yards)	416427.4	4376.4	5413965.0	705142.0
c_j	$1,922,517.34	$46,440.77	$14,477,345.28	$2,455,009.14
$e_w - b_w$ (days)	144.6	30.0	275.0	71.5
r_d (cubic yards per day)	14636.7	1237.7	66418.0	14584.4

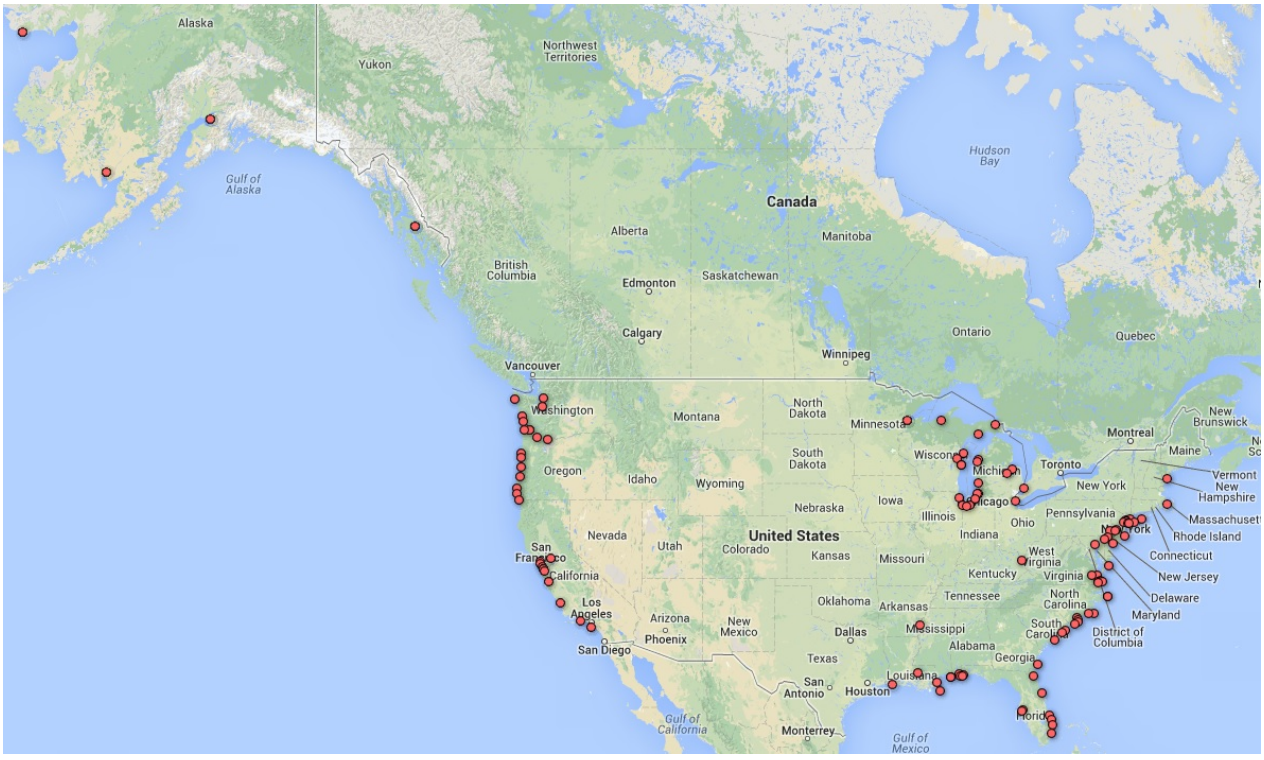

Figure 4.1: Graphical Depiction of 116 Dredge Project Locations

We identified 130 distinct restricted periods across the 116 jobs that are time windows when dredging is not permitted at certain job locations due to environmental concerns. Note that

some jobs may have none, one or multiple restricted periods. Production rates of dredge vessels in Table 4.1 reflect a historical statistical average of multiple dredging projects.

Table 4.2: Experiment Design

$\|J\|$	Number of RPs	$\|D\|$	Job Size (q_j)	Vessel Type
32	36	10, 15, 20, 25, 30	Original (O) Random (R)	Fastest (F) Random (R) Slowest (S)
57	63	10, 15, 20, 25, 30	Original (O) Random (R)	Fastest (F) Random (R) Slowest (S)
116	130	10, 15, 20, 25, 30	Original (O) Random (R)	Fastest (F) Random (R) Slowest (S)

Table 4.2 demonstrates the properties of 90 problem instances that were used to measure the performance of the solution techniques described in Section 4.2. The first ($|J| = 32$) and second ($|J| = 57$) sets of jobs do not have any common dredge projects; whereas, the final job set ($|J| = 116$) includes all job locations and restricted periods. For each problem instance, we either select the fastest, slowest or randomly picked distinct dredge vessels to build the fleet from a 40 vessel fleet. We also generate job sizes while keeping all other parameters constant with respect to a discrete uniform distribution in a range of $[10K, 2M]$. Thus, "32-36-10-O-F" stands for the problem instance with 32 jobs with Original job sizes, 36 restricted periods and the Fastest 10 dredge vessels of the fleet.

4.4 Computational Results

The parallel machine scheduling problem formulations in Section 4.2 are modeled in IBM ILOG CPLEX Optimization Studio 12.3 (IBM, 2011) which uses IBM ILOG CPLEX 12.3 to solve ILP, IBM ILOG CP Optimizer 12.3 to solve CP and both to solve logic based Benders algorithms. All formulations are modeled in C++ programming language. IBM ILOG Concert Technology is utilized for embedding the formulations in C++ language into IBM ILOG CPLEX and CP Optimizer. We run all test problems on a Core 2 Duo 2.93 GHz, 16 GB RAM computer. We create and solve the master problems from scratch with the updated Benders cuts at each iteration.

As mentioned before, we observe that for a medium size problem instance ($|D| = 10$ and $|J| = 32$), CPLEX cannot even begin to solve the (DS) and (DS-M) models presented in Section 4.2.1 due to the memory insufficiency. Therefore, we do not report any experiment results for models (DS) and (DS-M). Table 4.A.1 shows computational results obtained by solving the problem (CP-DS) and Benders decompositions algorithms (Benders 1 and 2). Note that (CP-DS) is able to report at least a feasible solution to all instances within a 30 minute time limit. Similarly, Benders 1 and 2 also terminate the search when the solution time reaches 30 minutes or the number of iterations is equal to 200. As seen in Table 4.A.1, decomposition algorithms are shown to be more efficient in proving optimality and effective in providing more optimal solutions within a reasonable amount of time and allowed iteration limit. Further statistics on the type of the final solution obtained by each method can be seen in Table 4.1. Although the Benders 1 algorithm increases the number of optimal solutions identified by CP for the problem instances with $|J| = 32$ and $|J| = 57$, it is outperformed by Benders 2 in terms of number of optimal solutions except for the problem instances with $|J| = 32$ and original job sizes. Moreover, Benders 2 is able to produce optimal solutions for the largest dataset with $|J| = 116$ while Benders 1 and (CP-DS) fail to provide any. Overall, 37 of the 45 problem instances with original job sizes and 22 out of 45 problem instances with randomly generated job sizes are solved to optimality by one of our methods and for the rest of them, (CP-DS) terminates with at least a feasible solution.

The reason that Benders 2 outperforms Benders 1 (and also (CP-DS)) in terms of finding optimal solutions for the problem instances with $|J| = 116$ is that as the number of projects increases, the Benders cuts (4.24) throughout the iterations increase the size of the master problem (M1-DS) of Benders 1 which eventually leads to large computation times. Hence, the time or iteration limit is reached before generating the essential cuts required to find the optimal solution. Benders 2, on the other hand, has a relatively larger subproblem compared to the ones in Benders 1 but it has a master problem with less constraints. One can assume that proving that the subproblem (S2-DS) of Benders 2 is feasible or infeasible requires larger computation time since it handles the assignment decisions for all dredge vessels as opposed to checking feasibility of each

Table 4.1: Number of Optimal (O), Feasible (F) and Infeasible (I) Solutions

q_j		CP			Benders 1			Benders 2			Overall		
Original	$\|J\|$	O	F	I	O	F	I	O	F	I	O	F	I
	32	4	11	0	12	0	3	9	0	6	14	1	0
	57	1	14	0	7	0	8	11	0	4	14	1	0
	116	0	15	0	0	0	15	9	0	6	9	6	0
	Av. Time	1.68	1800	-	543	-	1425	1.03	-	258	-	-	-
	Av. Itrs.	-	-	-	95	-	177	1	-	200	-	-	-
DU(10K, 2M)	$\|J\|$	O	F	I	O	F	I	O	F	I	O	F	I
	32	2	13	0	5	0	10	6	0	9	10	5	0
	57	1	14	0	1	0	14	9	0	6	10	5	0
	116	0	15	0	0	0	15	2	0	13	2	13	0
	Av. Time	0.78	1800	-	846	-	1241	1.46	0	584	-	-	-
	Av. Itrs.	-	-	-	121	-	190	1	0	157	-	-	-
	Total	**8**	**82**	**0**	**25**	**0**	**65**	**46**	**0**	**44**	**59**	**31**	**0**

dredge vessel in a separate problem in (S1-DS). The results in Tables 4.A.1 and 4.1 suggest that solving (S2-D2) does not require more computational time since the iteration limit is generally the termination criteria in most problem instances. When the iteration upper limit or time limit is not reached, Benders 2 is able to explore the optimal solution in at most one iteration. This situation exemplifies the high efficiency of CP in obtaining feasible/infeasible solutions for highly constrained problems.

In order to assess the quality of the feasible solutions obtained by (CP-DS) for the problem instances that are solved to optimality, we calculate the optimality gap % such that

$$\text{Optimality Gap } \% \ = 100 * \left(\frac{\text{Optimal Objective} - \text{CP Objective}}{\text{Optimal Objective}} \right).$$

Figure 4.1 plots the optimality gap of (CP-DS) for the problem instances that are solved to optimality by at least one of the methods. An interesting observation is that there are 31 problem instances with 0% optimality gap. This means that CP is able to explore 31 optimal solutions but it is only able to prove the optimality of eight of them as listed in Table 4.A.1. Figure 4.1 also shows that the quality of the feasible solutions obtained by (CP-DS) is impressive since the largest optimality gap of (CP-DS) is 3%.

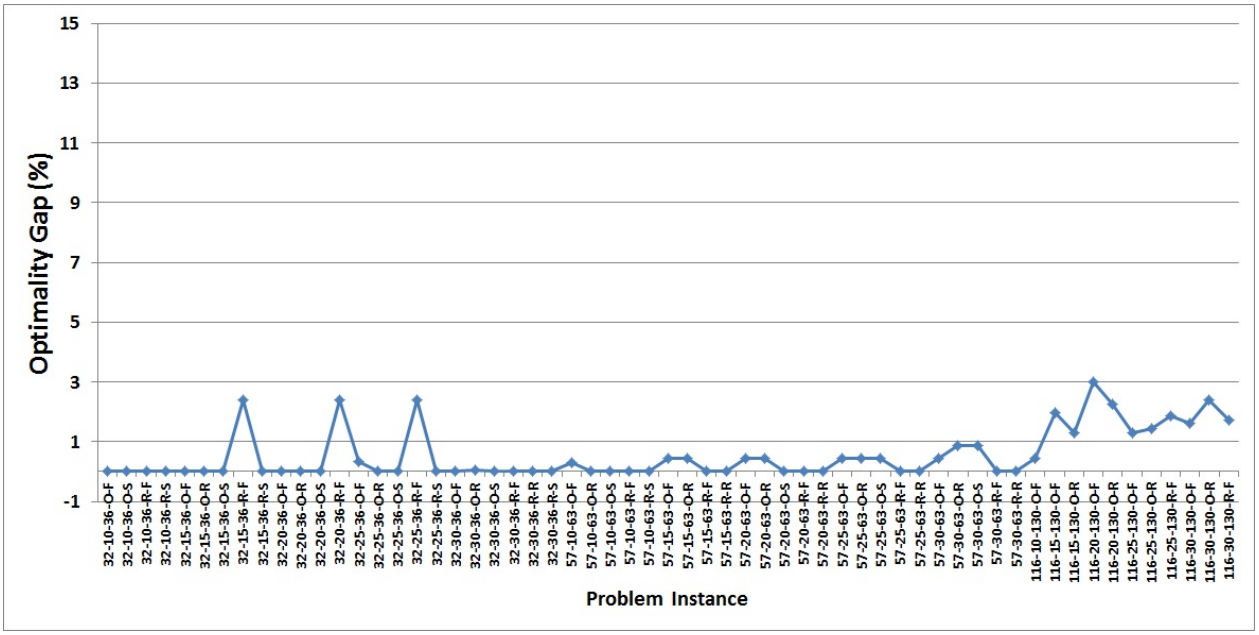

Figure 4.1: Optimality Gap of (CP-DS)

For the problem instances that are not solved to optimality, the gap between feasible (CP-DS) solution and the last infeasible solutions of the Benders algorithms increases dramatically as the job size increases. However, these gaps are smaller between Benders 1 and (CP-DS) than the ones between Benders 2 and (CP-DS). This is because the cuts generated in Benders 1 convey the restrictions on job assignments to dredge equipments to the master problem whereas the cuts in Benders 2 only restricts whether to process a job or not and do not inform the master problem on further infeasible assignments. Therefore, Benders 1 is able to prevent the infeasible job assignments to specific dredge equipments for further iterations. However, this advantage is not good enough to provide more optimal solutions than Benders 2 as highlighted in Table 4.1.

The results obtained by the (CP-DS-M) model to minimize the makespan on problem instances in Table 4.2 are listed in Table 4.2. The minimum makespan (v) value found within 30 minutes time limit is reported if the problem is not infeasible (I). Furthermore, total dredge, travel and idle days are illustrated for each problem instance. Note that CP is able to identify the optimal solutions for the problem instances that are not infeasible within the time limit. Especially for problem instances with $|J| = 116$, most of the problems are proven to be infeasible in a very

short time, and in the ones that are feasible, the optimal solutions are obtained. The largest solution time is 166.99 seconds for the problem instances that are solved to optimality by (CP-DS-M) model. The only time CP could not prove optimality is for the problem instance "57-15-63-R-R". Other highlighted problem instances for which CP reaches the time limit are the ones where a feasible solution is not identified within 30 minutes. Total travel, dredge and idle time of the fleet for each problem instance can be seen in Table 4.2. Idle time reported in Table 4.2 only accounts for the idle time of a dredge vessel that handles at least one project. If a dredge vessel has no jobs assigned to it, it can be removed from the fleet and assigned to some other operations. Also, the calendar days before and after a particular dredge is utilized are not reported as idle time.

Table 4.2: Experiment Results for (CP-DS-M) Model with Minimizing Makespan Objective

Problem	v	Time	Dredge Time	Travel Time	Idle Time	Problem	v	Time	Dredge Time	Travel Time	Idle Time	Problem	v	Time	Dredge Time	Travel Time	Idle Time
32-10-36-O-F	336	1.23	601	442	524	57-10-63-O-F	290	1.24	577	596	774	116-10-130-O-F	336	13.42	1434	902	557
32-10-36-O-R	I	0.42	-	-	-	57-10-63-O-R	300	1.42	1411	500	506	116-10-130-O-R	I	0.38	-	-	-
32-10-36-O-S	I	0.43	-	-	-	57-10-63-O-S	I	0.40	-	-	-	116-10-130-O-S	I	0.24	-	-	-
32-10-36-R-F	364	1.11	1033	724	404	57-10-63-R-F	298	3.23	1977	835	24	**116-10-130-R-F**	**I**	**1800**	**-**	**-**	**-**
32-10-36-R-R	I	0.35	-	-	-	57-10-63-R-R	I	0.99	-	-	-	116-10-130-R-R	I	0.39	-	-	-
32-10-36-R-S	I	0.41	-	-	-	57-10-63-R-S	I	0.34	-	-	-	116-10-130-R-S	I	0.33	-	-	-
32-15-36-O-F	336	1.17	666	883	639	57-15-63-O-F	290	1.48	693	927	729	116-15-130-O-F	336	3.76	2067	1863	266
32-15-36-O-R	336	1.20	1250	409	804	57-15-63-O-R	290	1.47	1253	600	700	116-15-130-O-R	336	166.99	2593	1555	373
32-15-36-O-S	I	0.36	-	-	-	57-15-63-O-S	I	0.27	-	-	-	116-15-130-O-S	I	0.39	-	-	-
32-15-36-R-F	364	0.88	1377	657	336	57-15-63-R-F	298	1.25	2261	667	244	**116-15-130-R-F**	**I**	**1800**	**-**	**-**	**-**
32-15-36-R-R	I	0.19	-	-	-	**57-15-63-R-R**	**323**	**1800**	**3238**	**724**	**55**	116-15-130-R-R	I	0.37	-	-	-
32-15-36-R-S	I	0.28	-	-	-	57-15-63-R-S	I	0.23	-	-	-	116-15-130-R-S	I	0.34	-	-	-
32-20-36-O-F	336	0.87	733	432	761	57-20-63-O-F	290	1.48	825	941	1023	116-20-130-O-F	336	4.07	2053	1412	1363
32-20-36-O-R	336	0.94	945	842	611	57-20-63-O-R	290	0.84	1298	1057	866	116-20-130-O-R	336	6.14	4461	1704	302
32-20-36-O-S	I	0.29	-	-	-	57-20-63-O-S	I	0.31	-	-	-	116-20-130-O-S	I	0.41	-	-	-
32-20-36-R-F	364	1.14	1471	777	489	57-20-63-R-F	298	1.51	2856	1012	382	**116-20-130-R-F**	**I**	**1800**	**-**	**-**	**-**
32-20-36-R-R	I	0.21	-	-	-	57-20-63-R-R	298	123.98	3381	507	50	**116-20-130-R-R**	**I**	**1800**	**-**	**-**	**-**
32-20-36-R-S	I	0.26	-	-	-	57-20-63-R-S	I	0.31	-	-	-	116-20-130-R-S	I	0.24	-	-	-
32-25-36-O-F	336	1.15	872	714	722	57-25-63-O-F	290	1.63	917	939	1181	116-25-130-O-F	336	3.01	2444	1525	1204
32-25-36-O-R	336	1.05	1230	243	506	57-25-63-O-R	290	1.60	1525	1057	726	116-25-130-O-R	336	4.13	5135	1879	549
32-25-36-O-S	I	0.29	-	-	-	57-25-63-O-S	I	0.37	-	-	-	116-25-130-O-S	I	0.45	-	-	-
32-25-36-R-F	364	0.75	1763	157	646	57-25-63-R-F	298	1.56	3153	1077	656	116-25-130-R-F	364	59.17	6447	1731	301
32-25-36-R-R	I	0.41	-	-	-	57-25-63-R-R	298	3.14	4179	732	633	**116-25-130-R-R**	**I**	**1800**	**-**	**-**	**-**
32-25-36-R-S	I	0.29	-	-	-	57-25-63-R-S	I	0.34	-	-	-	116-25-130-R-S	I	0.39	-	-	-
32-30-36-O-F	336	1.12	892	850	432	57-30-63-O-F	290	1.85	1122	783	1393	116-30-130-O-F	336	3.78	2601	1514	1816
32-30-36-O-R	336	0.97	1035	790	474	57-30-63-O-R	290	1.62	1538	1176	548	116-30-130-O-R	336	4.73	5041	1515	674
32-30-36-O-S	I	0.29	-	-	-	57-30-63-O-S	332	1.88	2420	1060	943	116-30-130-O-S	I	0.44	-	-	-
32-30-36-R-F	364	0.77	2120	613	931	57-30-63-R-F	298	1.51	3548	847	861	116-30-130-R-F	364	4.30	8183	2045	332
32-30-36-R-R	364	0.79	2318	268	485	57-30-63-R-R	298	1.73	4704	786	426	**116-30-130-R-R**	**I**	**1800**	**-**	**-**	**-**
32-30-36-R-S	I	0.32	-	-	-	57-30-63-R-S	I	0.40	-	-	-	116-30-130-R-S	I	0.45	-	-	-

4.5 Concluding Remarks

In this study, we discuss the advantages and disadvantages of three modeling approaches for the parallel machine scheduling problem with sequence dependent setup times, time windows, dissimilar dredge equipments and non-identical job durations. We propose three different solution methods: (i) integer programming, (ii) constraint programming, and (iii) logic-based Benders decomposition algorithms to approach this problem with a *maximizing total profit* objective function. We also propose a CP formulation for the same problem with a *minimizing makespan* objective function.

We prepare real-life test instances in collaboration with the USACE to be able to optimize the inland waterway infrastructure maintenance operations to make the waterways navigable for the supply chain activities. For most of the problem instances, we observe that the two proposed Benders algorithms are much more efficient and effective than either pure ILP or CP. Collectively, CP and Benders 1 and 2 are able to obtain 59 optimal solutions out of 90 problem instances with original and randomly generated job sizes ($q_j \sim DU(10K, 2M)$). Based on our further computational experiments, we observe that the number of optimal solutions for the 45 problem instances with randomly generated job sizes ($q_j \sim DU(10K, 4M)$) decreases to 10, whereas the number of optimal solutions for the problem instances with the original job sizes remains at the same level (37).

We also develop the (CP-DS-M) model for the same problem with a *minimizing makespan* objective. The computational results show that it has the capability of achieving optimal solutions in a very short time unless the problem is infeasible. Infeasible problems are also detected in a reasonable amount of time. For this purpose, possible decomposition algorithms for the same problem with minimizing makespan objective are not considered.

Appendix

4.A Detailed Experiment Results for the Problem with Maximizing Profit Objective

Problem	(CP-DS)		Benders 1			Benders 2		
	Time	Status	Time	Iteration	Status	Time	Iteration	Status
32-10-36-O-F	1800	F	485.9	200	I	0.8	1	O
32-10-36-O-R	1800	F	514.1	200	I	234.6	200	I
32-10-36-O-S	1.007	O	431.3	153	O	225.3	200	I
32-10-36-R-F	1800	F	524.5	200	I	0.9	1	O
32-10-36-R-R	1800	F	611.4	200	I	236.3	200	I
32-10-36-R-S	0.776	O	559.1	200	I	235.4	200	I
32-15-36-O-F	1800	F	114.4	39	O	0.8	1	O
32-15-36-O-R	1800	F	57.0	20	O	0.8	1	O
32-15-36-O-S	1.041	O	1057.5	200	I	236.3	200	I
32-15-36-R-F	1800	F	810.2	200	I	0.9	1	O
32-15-36-R-R	1800	F	760.0	200	I	237.2	200	I
32-15-36-R-S	0.848	O	940.1	200	I	234.6	200	I
32-20-36-O-F	1800	F	341.3	80	O	0.8	1	O
32-20-36-O-R	1800	F	182.5	47	O	0.7	1	O
32-20-36-O-S	0.686	O	1087.4	189	O	234.5	200	I
32-20-36-R-F	1800	F	848.1	152	O	0.9	1	O
32-20-36-R-R	1800	F	824.1	200	I	236.4	200	I
32-20-36-R-S	1800	F	1232.1	200	I	236.8	200	I
32-25-36-O-F	1800	F	320.5	84	O	0.8	1	O
32-25-36-O-R	1800	F	149.3	38	O	0.8	1	O
32-25-36-O-S	1.111	O	220.4	50	O	234.9	200	I
32-25-36-R-F	1800	F	349.3	68	O	0.9	1	O
32-25-36-R-R	1800	F	1006.2	200	I	236.7	200	I
32-25-36-R-S	1800	F	843.0	137	O	234.8	200	I
32-30-36-O-F	1800	F	451.1	71	O	0.8	1	O
32-30-36-O-R	1800	F	178.6	41	O	0.8	1	O
32-30-36-O-S	1800	F	421.9	64	O	238.0	200	I
32-30-36-R-F	1800	F	959.9	123	O	1.0	1	O
32-30-36-R-R	1800	F	1283.7	200	I	0.9	1	O
32-30-36-R-S	1800	F	880.3	107	O	260.4	200	I
57-10-63-O-F	1800	F	1014.2	200	I	1.0	1	O
57-10-63-O-R	1800	F	1552.7	200	I	1.0	1	O
57-10-63-O-S	4.24	O	1800.0	126	I	305.8	200	I
57-10-63-R-F	1800	F	737.2	200	I	1.3	1	O
57-10-63-R-R	1800	F	575.2	200	I	387.3	200	I
57-10-63-R-S	0.715	O	673.5	200	I	249.5	200	I
57-15-63-O-F	1800	F	135.8	35	O	0.9	1	O
57-15-63-O-R	1800	F	1800.0	14	I	0.9	1	O
57-15-63-O-S	1800	F	1515.1	200	I	271.9	200	I
57-15-63-R-F	1800	F	1353.3	200	I	1.2	1	O

Problem	(CP-DS)		Benders 1			Benders 2		
	Time	Status	Time	Iteration	Status	Time	Iteration	Status
57-15-63-R-R	1800	F	955.8	200	I	2.9	1	O
57-15-63-R-S	1800	F	1083.5	200	I	226.0	200	I
57-20-63-O-F	1800	F	1419.6	200	I	1.2	1	O
57-20-63-O-R	1800	F	1037.4	200	I	0.5	1	O
57-20-63-O-S	1800	F	1166.0	144	O	272.7	200	I
57-20-63-R-F	1800	F	1124.4	200	I	1.2	1	O
57-20-63-R-R	1800	F	1048.6	200	I	0.7	1	O
57-20-63-R-S	1800	F	1494.1	200	I	143.8	200	I
57-25-63-O-F	1800	F	314.3	110	O	0.6	1	O
57-25-63-O-R	1800	F	1010.9	200	I	0.6	1	O
57-25-63-O-S	1800	F	1682.5	200	O	273.3	200	I
57-25-63-R-F	1800	F	1036.4	200	I	0.8	1	O
57-25-63-R-R	1800	F	1282.5	200	I	1.0	1	O
57-25-63-R-S	1800	F	1800.0	161	I	253.5	200	I
57-30-63-O-F	1800	F	1074.8	177	O	1.0	1	O
57-30-63-O-R	1800	F	477.8	72	O	0.9	1	O
57-30-63-O-S	1800	F	1510.1	190	O	1.0	1	O
57-30-63-R-F	1800	F	1198.7	140	O	1.3	1	O
57-30-63-R-R	1800	F	1448.6	200	I	0.9	1	O
57-30-63-R-S	1800	F	1800.0	150	I	256.4	200	I
116-10-130-O-F	1800	F	1021.5	200	I	1.1	1	O
116-10-130-O-R	1800	F	1130.1	200	I	291.1	200	I
116-10-130-O-S	1800	F	899.3	200	I	287.0	200	I
116-10-130-R-F	1800	F	819.9	200	I	1800.0	2	I
116-10-130-R-R	1800	F	832.6	200	I	308.4	200	I
116-10-130-R-S	1800	F	881.1	200	I	305.7	200	I
116-15-130-O-F	1800	F	1800.0	191	I	1.7	1	O
116-15-130-O-R	1800	F	1796.9	200	I	1.6	1	O
116-15-130-O-S	1800	F	1663.5	200	I	288.6	200	I
116-15-130-R-F	1800	F	1732.1	200	I	1800.0	2	I
116-15-130-R-R	1800	F	1463.3	200	I	312.5	200	I
116-15-130-R-S	1800	F	1605.4	200	I	312.5	200	I
116-20-130-O-F	1800	F	1800.0	182	I	1.6	1	O
116-20-130-O-R	1800	F	1800.0	177	I	1.6	1	O
116-20-130-O-S	1800	F	1800.0	177	I	290.3	200	I
116-20-130-R-F	1800	F	1800.0	194	I	1800.0	2	I
116-20-130-R-R	1800	F	1800.0	183	I	1800.0	2	I
116-20-130-R-S	1800	F	1800.0	189	I	173.1	200	I
116-25-130-O-F	1800	F	1359.2	200	I	0.9	1	O
116-25-130-O-R	1800	F	1577.0	200	I	1.0	1	O
116-25-130-O-S	1800	F	1800.0	185	I	165.1	200	I
116-25-130-R-F	1800	F	1691.4	200	I	5.9	1	O
116-25-130-R-R	1800	F	1800.0	191	I	1800.0	2	I
116-25-130-R-S	1800	F	1800.0	159	I	175.5	200	I
116-30-130-O-F	1800	F	1800.0	148	I	1.9	1	O
116-30-130-O-R	1800	F	1800.0	50	I	1.8	1	O

Problem	(CP-DS)		Benders 1			Benders 2		
	Time	Status	Time	Iteration	Status	Time	Iteration	Status
116-30-130-O-S	1800	F	1800.0	144	I	292.1	200	I
116-30-130-R-F	1800	F	1800.0	149	I	2.4	1	O
116-30-130-R-R	1800	F	1800.0	133	I	1800.0	2	I
116-30-130-R-S	1800	F	1800.0	127	I	311.4	200	I

Table 4.A.1: Experiment Results for the Problem with Maximizing Profit Objective

4.B Work Verification Letter

UNIVERSITY OF ARKANSAS

College of Engineering
Department of Industrial Engineering

Date: July 8, 2014

Graduate School
University of Arkansas

Dear Dr. Needy:

I am writing to verify that Ridvan Gedik completed more than 51% of the work for the chapter titled "Analysis of a Parallel Machine Scheduling Problem with Sequence Dependent Setup Times and Time Windows," in his dissertation. He is also the first author of this article.

Sincerely,

Chase Rainwater
cer@uark.edu
479-575-2687
Assistant Professor
Department of Industrial Engineering
University of Arkansas

Bibliography

Allahverdi, A., Ng, C., Cheng, T. E., and Kovalyov, M. Y. (2008). A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, 187(3):985–1032.

Arkin, E. M. and Silverberg, E. B. (1987). Scheduling jobs with fixed start and end times. *Discrete Applied Mathematics*, 18(1):1 – 8.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik*, 4(1):238–252.

Cakici, E. and Mason, S. J. (2007). Parallel machine scheduling subject to auxiliary resource constraints. *Production Planning and Control*, 18(3):217–225.

Cesta, A., Oddi, A., and Smith, S. (2002). A constraint-based method for project scheduling with time windows. *Journal of Heuristics*, 8:109–136.

Dickerson, D., Reine, K. J., and Clarke, D. G. (I998). Economic impacts of environmental windows associated with dredging operations. Technical report, U.S. Army Engineer Research and Development Center, Vicksburg, MS, www.wes.army.mil/el/dots/doer.

Edis, E. B. and Oguz, C. (2012). Parallel machine scheduling with flexible resources. *Computers & Industrial Engineering*, 63(2):433 – 447.

Edis, E. B. and Ozkarahan, I. (2011). A combined integer/constraint programming approach to a resource-constrained parallel machine scheduling problem with machine eligibility restrictions. *Engineering Optimization*, 43(2):135–157.

Focacci, F., Lodi, A., and Milano, M. (2002). Mathematical programming techniques in constraint programming: A short overview. *Journal of Heuristics*, 8:7–17.

Harjunkoski, I. and Grossmann, I. E. (2002). Decomposition techniques for multistage scheduling problems using mixed-integer and constraint programming methods. *Comp. Chem. Engng*, 26:1533–1552.

He, F. and Qu, R. (2012). A constraint programming based column generation approach to nurse rostering problems. *Computers & Operations Research*, 39(12):3331 – 3343.

Heipcke, S. (1999). Comparing constraint programming and mathematical programming approaches to discrete optimisation-the change problem. *The Journal of the Operational Research Society*, 50(6):pp. 581–595.

Hooker, J. (2002). Logic, optimization, and constraint programming. *INFORMS Journal on Computing*, 14:295–321.

Hooker, J. (2006a). *Handbook of Constraint Programming*, pages 205–239. Elsevier.

Hooker, J. N. (2006b). *Integrated Methods for Optimization (International Series in Operations Research & Management Science)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Hooker, J. N. (2007a). Good and bad futures for constraint programming (and operations research). *Constraint Programming Letters*, 1:21 – 32.

Hooker, J. N. (2007b). Planning and scheduling by logic-based benders decomposition. *Operations Research*, 55(3):588–602.

Hooker, J. N., Ottosson, G., Thorsteinsson, E. S., and Kim, H.-J. (1999). On integrating constraint propagation and linear programming for combinatorial optimization.

IBM (2011). Ibm ilog cplex optimization studio v12.3.

Jain, V. and Grossmann, I. E. (2001). Algorithms for hybrid milp/cp models for a class of optimization problems. *INFORMS Journal on Computing*, 13:258–276.

Lombardi, M. and Milano, M. (2012). Optimal methods for resource allocation and scheduling: a cross-disciplinary survey. *Constraints*, 17:51–85.

Lund, J. (1990). Scheduling maintenance dredging on single reach with uncertainty. *Journal of Waterway, Port, Coastal, and Ocean Engineering*, 116(2):211–231.

Menon, H. and Lansey, K. (1990). Maintenance scheduling for water resource systems: An application to advance maintenance dredging. Proceedings of the 1990 National Conference, pages 575–579. American Society of Civil Engineers.

Milano, M. (2004). *Constraint and Integer Programming: Toward a Unified Methodology*. Operations Research/Computer Science Interfaces Series. Springer.

Mönch, L., Fowler, J. W., Dauzère-Pérès, S., Mason, S. J., and Rose, O. (2011). A survey of problems, solution techniques, and future challenges in scheduling semiconductor manufacturing operations. *Journal of Scheduling*, 14(6):583–599.

Pearn, W. L., Chung, S. H., and Yang, M. H. (2002). The wafer probing scheduling problem (wpsp). *The Journal of the Operational Research Society*, 53(8):pp. 864–874.

Pointon, M. (1996). Dredging costs analysis information paper. Technical report, sponsored by the U.S. Army Corps of Engineers Institute for Water Resources, Alexandria, VA,, http://www.wrcndc.itsace.army.mil.

Ratick, S. and Garriga, H. (1996). Risk-based spatial decision support system for maintenance dredging of navigation channels. *Journal of Infrastructure Systems*, 2(1):15–22.

Rodosek, R., Wallace, M., and Hajian, M. (1999). A new approach to integrating mixed integer programming and constraint logic programming. *Annals of Operations Research*, 86:63–87.

Rojanasoonthon, S. and Bard, J. (2005). A grasp for parallel machine scheduling with time windows. *INFORMS Journal on Computing*, 17(1):32–51.

Sadykov, R. and Wolsey, L. A. (Spring 2006). Integer programming and constraint programming in solving a multimachine assignment scheduling problem with deadlines and release dates. *INFORMS Journal on Computing*, 18(2):209–217.

Topaloglu, S. and Ozkarahan, I. (2011). A constraint programming-based solution approach for medical resident scheduling problems. *Computers & Operations Research*, 38(1):246 – 255.

Trick, M. A. and Yildiz, H. (2011). Benders' cuts guided large neighborhood search for the traveling umpire problem. *Naval Research Logistics (NRL)*, 58(8):771–781.

Türsel Eliiyi, D. and Azizoğlu, M. (2009). A fixed job scheduling problem with machine-dependent job weights. *International Journal of Production Research*, 47(9):2231–2256.

van Hoeve, W.-J. and Katriel, I. (2006). *Handbook of Constraint Programming*. Elsevier.

5. CONCLUSION

In this dissertation, we propose novel solution approaches for two real life and highly complex scheduling problems. We first introduced a patient scheduling problem with deterministic patient arrivals to the proton therapy facility in Chapter 2. Single and multi-criteria linear programming models were proposed in order to assess the impacts of several operational restrictions. A notable feature of these models was ability to account for the deviation from a predetermined strategic level patient mix restrictions for each patient category, which has received little attention in the literature. We exploited the mathematical structures of the model with strict patient mix constraints and derived analytical equations for the optimal solution under several operational restrictions. These efforts led to a set of rule of thumbs that can be utilized to assess the impacts of several input parameters and patient mix levels on the capacity utilization without solving optimization problems. We also explored the necessary and sufficient conditions in order to be able to generate efficient frontiers of the bicriteria problem without any additional side constraint analytically. The computational study of our algorithm, which relies on parametric perturbation analysis and optimal solution characteristics, demonstrated the efficiency gains in solving this problem by our approach. A brief discussion regarding the possible ways of analytically generating efficient frontiers for the same problem with some of the side constraints was provided.

In Chapter 3, we investigated possible solution techniques to the patient scheduling problem introduced in Chapter 2 with stochastic patient arrivals. We proposed an MDP model that seeks to obtain the best patient admission policies while maximizing the total expected number of treatment sessions and penalizing the deviation from the patient mix restrictions. It provided very useful insights in determining the best patient admission policies in the case of an unexpected opening in the facility (i.e. no-shows, appointment cancellations etc.). However, our numerical experiments illustrated that the MDP state space grows exponentially and it becomes computationally intractable as the size of input parameters (number of available time slots, patient categories etc.) increases. Thus, we developed an aggregate MDP model that is able to approximate optimal patient admission policies by fixed-weight aggregation technique. Our further experi-

ments revealed that it is capable of obtaining high quality executable approximate policies for the realistic problem instances.

Finally, in Chapter 4, we dealt with a different version of the parallel machine scheduling problem with sequence dependent setup times and time windows. We studied this problem with two different objective functions: (i) maximizing the total reward achieved and (ii) minimizing the makespan. For a given time horizon, our objective is to find the best assignment scheme of nonidentical jobs to dissimilar machines. In addition, we accounted for the sequence dependent setup time in between consecutive jobs assigned to the same machine and (multiple) restricted time periods. We realized that the proposed ILP model fails to solve even very small problem instances. Thus, we developed pure CP models for the same problem with two different objectives. Our computational efforts demonstrated that the pure CP performs very well on the problem with minimizing makespan objective. However, although it finds feasible solutions very quickly for the problem with maximizing total profit objective, it fails to terminate with the optimal solution on most of the instances. In order to address this issue, we developed two different novel logic-based Benders decomposition algorithms that are capable of locating more optimal solutions.